R and D CENTER LABORATORY

TECHNICAL REPORT

NO. 12590

GERERALIZED COORDINATE PARTITIONING IN DYNAMIC ANALYSIS OF MECHANICAL SYSTEMS

Final Report

16 June 1981

Contract No: DAAK30-78-C-0096
DAAG29-79-C-0221

R.A. Wehage and E.J. Haug
College of Engineering,
University of Iowa
University of Iowa Report No. 75

by Ronald R. Beck, Project Engineer, TACOM

U.S. ARMY TANK-AUTOMOTIVE COMMAND
RESEARCH AND DEVELOPMENT CENTER
Warren, Michigan 48090

DISCLAIMER NOTICE

THIS DOCUMENT IS BEST QUALITY PRACTICABLE. THE COPY FURNISHED TO DTIC CONTAINED A SIGNIFICANT NUMBER OF PAGES WHICH DO NOT REPRODUCE LEGIBLY.

NOTICES

The findings in this report are not to be construed as an official Department of the Army position.

Mention of any trade names or manufacturers in this report shall not be construed as advertising nor as an official endorsement of approval of such products or companies by the U.S. Government.

Destroy this report when it is no longer needed. Do not return it to originator.

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER: 12590	2. GOVT ACCESSION NO.: AD-A108 683
	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle): Generalized Coordinate Partitioning in Dynamic Analysis of Mechanical Systems	5. TYPE OF REPORT & PERIOD COVERED: Final, Dec. 1980
	6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s): R.A. Wehage and E.J. Haug University of Iowa Ronald R. Beck, TACOM	8. CONTRACT OR GRANT NUMBER(s): DAAK30-78-C-0096 DAAG29-79-C-0221
9. PERFORMING ORGANIZATION NAME AND ADDRESS: The University of Iowa College of Engineering Iowa City, IA 52242	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS: U.S. Army Tank-Automotive Command, R&D Center Tank-Automotive Concepts Lab, DRSTA-ZSA Warren, MI 48090	12. REPORT DATE: 16 June 1981 13. NUMBER OF PAGES: 205
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)	15. SECURITY CLASS. (of this report): Unclassified 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Matrices, Sparse Matrix, Dynamics, Bodies Response Control, Interaction, Stability, Lagrangian Functions

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A computer-based method for formulation and efficient solution of nonlinear, constrained differential equations of motion is developed for planar mechanical systems. Nonlinear holonomic constraint equations and differential equations of motion are written in terms of a maximal set of Cartesian generalized coordinates, to facilitate the general formulation of constraints and forcing functions. A Gaussian elimination algorithm with full pivoting decomposes the constraint Jacobian matrix, identifies dependent variables, and constructs an influence coefficient matrix relating variations in dependent and independent variables.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

This information is employed to numerically construct a reduced system of differential equations whose solution yields the total system dynamic response. A numerical integration algorithm with positive-error control, employing a predictor-corrector algorithm with variable order and step-size, is developed that integrates for only the independent variables, yet effectively determines dependent variables.

A general method is developed for dynamic analysis of systems with impulsive forces, impact, discontinuous constraints, and discontinuous velocities. This class of systems includes discontinuous kinematic and geometric constraints that characterize backlash and impact within systems. A method of computer generation of the impulse-momentum relations that define jump discontinuities in system velocity for large scale systems is developed. An event predictor employing logical functions of system state and working in conjunction with the new numerical integration algorithm efficiently controls its progress and allows for automatic equation reformulation.

Numerical results are presented for planar motion of a tracked articulated vehicular system with twenty-four degrees of freedom. A mechanism with discontinuous forces and velocities is simulated to demonstrate program generality and improved efficiency over previous modeling methods. An order of magnitude improvement in efficiency has been demonstrated.

ABSTRACT

A computer-based method for formulation and efficient solution of nonlinear, constrained differential equations of motion is developed for planar mechanical systems. Nonlinear holonomic constraint equations and differential equations of motion are written in terms of a maximal set of Cartesian generalized coordinates, to facilitate the general formulation of constraints and forcing functions. A Gaussian elimination algorithm with full pivoting decomposes the constraint Jacobian matrix, identifies dependent variables, and constructs an influence coefficient matrix relating variations in dependent and independent variables. This information is employed to numerically construct a reduced system of differential equations whose solution yields the total system dynamic response. A numerical integration algorithm with positive-error control, employing a predictor-corrector algorithm with variable order and step-size, is developed that integrates for only the independent variables, yet effectively determines dependent variables.

A general method is developed for dynamic analysis of systems with impulsive forces, impact, discontinuous constraints, and discontinuous velocities. This class of systems includes discontinuous kinematic and geometric constraints that characterize backlash and impact within systems. A method of computer generation of the

impulse-momentum relations that define jump discontinuities in system velocity for large scale systems is developed. An event predictor employing logical functions of system state and working in conjunction with the new numerical integration algorithm efficiently controls its progress and allows for automatic equation reformulation.

Numerical results are presented for planar motion of a tracked articulated vehicular system with twenty-four degrees of freedom. A mechanism with discontinuous forces and velocities is simulated to demonstrate the capabilities of the method. The examples were selected to demonstrate program generality and improved efficiency over previous modeling methods. An order of magnitude improvement in efficiency has been demonstrated.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS

Symbol	
A	Composite vector of A^i
A^i	Velocity dependent terms for the i^{th} body resulting from differentiating T^i with respect to t and q^i
A^u, A^v	Partitioning of A according to u and v
A_1	Actuator area - rod end
A_2	Actuator area - cylinder end
Bi	Rigid body numbers
B_f	Constant front buffer force
B_r	Constant rear buffer force
C^1	Angular, centripetal, and Coriolis acceleration due to independent constraints
C^2	Angular, centripetal, and Coriolis acceleration due to redundant constraints
c_{ij}	Damping Coefficient
dof	System degree of freedom
D_{w_i}	Damping force developed on i^{th} wheel due to interaction between wheel and terrain
d_1, d_2	Upper and lower lengths of the control stick
e	Coefficient of restitution
E(s)	Error signal voltage in control system
f	Vector of functions
f(t)	Impulsive force acting at contacting surfaces during impact

Symbol

F'	Force developed in successive track segments after being reduced by friction forces .
f_B	Functional relation between pin P and barrel B in weapon system
F_b	Constant force driving the barrel rearward
F_d	Vehicle suspension damping force function of $(-\dot{\ell}_{ij})$
F_f	Constant force drivi · barrel forward
F_f'	Friction force acting tangent to the wheel-terrain contact surface where $\|F_f'\| < \|F_f\|$ when the wheel does not slide
F_{fx} F_{fy}	Projection of frictional force F_f along x and y axes respectively
$\vec{F}_{ij}$	Force vector in the spring-damper-actuator
F_n	Normal force acting between roadwheel and terrain surface
F_{nx}, F_{ny}	Projection of F_n along x and y axes respectively
f_R	Functional relation between pin P and receiver R in weapon system
$F_R(s)$	Operator supplied fore and aft force to the control stick
F_s (F_{w_i})	Total vertical force acting upward on a wheel at the point of wheel-terrain contact
$F_s(\ell_{O_{ij}} - \ell_{ij})$	Vehicle suspension force function
F^u, F^v	Partitioning of generalized constraint reaction forces according to u and v
F_{w_i} (F_s)	Force developed on a wheel due to interaction between wheel and terrain
$F_{x_{ij}}$, $F_{y_{ij}}$	Projection of F_{ij} on the x and y axes respectively

Symbol	
$F_y(s)$	Pump yoke limiting force
$F_{O_{ij}}$	Actuator force applied along the spring-damper-actuator element
g	Vectors of functions
$g(x)$	A function that defines terrain or wheel trajectory elevation as a function of the wheel's global x position
h	Time step in the multistep integration formulas
$h(x)$	A function that defines the angular position of a vector passing through the point of wheel-terrain contact and center of wheel
H	Influence coefficient matrix
$\hat{H}$	The matrix H evaluated at points in (τ_1, τ_2)
$\vec{i}, \vec{j}$	Unit vectors parallel to the x and y axes, respectively
I	Identity matrix
$\bar{I}$	Rectangular matrix of full row rank consisting of a null and I matrix when columns are permuted according to u and v
$I(s)$	Amplifier current
J	Jacobian matrix defined in Equation 2.33
J_i	Centroidal polar moment of inertia of the i^{th} body
k	Degree of a polynomial in the multistep integration formula
k'	Number of logical times
K_d, K_v	Constants that characterize a vehicle power train
K_i	Measure of velocity ratios
k_{ij}	Spring constant

Symbol	
K_{RP}	Rotational potentiometer gain
K_v	Servo valve flow gradient
K_1, K_2	Amplifier gain
K_1, K_2	Springs attached to the lower part of the control stick
K_3	Pump stroke mechanism velocity gradient
K_4	Actuator potentiometer feedback gain
L	Combined system leakage coefficient
L, U	Lower and upper triangular matrices. In permuted form $L \cdot U = \Phi_u$
ℓ_{ij}	Deformed spring length
ℓ^j	The j^{th} logical variable
$\ell_{O_{23}}$	Reference hydraulic actuator length
ℓ_{23}	Total hydraulic actuator length
$\ell_{O_{ij}}$	Undeformed spring length
LR, UR	Residual lower and upper matrices. In permuted form $L \cdot UR = \Phi_v$
m	Number of constraint equations
M	Composite system mass matrix
m'	Number of logical functions
M^i	Mass matrix of the i^{th} body
m_i	Mass of the i^{th} body
M^{uu}, M^{uv} } M^{vu}, M^{vv} }	Partitioning of the mass matrix according to u and v
n	Number of system rigid bodies

Symbol	
$\bar{n}$	Row vector that determines the relative velocity direction in an internal impact
N	Row vector that relates generalized velocities $\dot{q}$ during internal impact
N^u, N^v	Partitioning of N according to u and v
p	Magnitude of impulse acting at contacting surfaces during impact
P	Vector of generalized impulses
P(s)	System pressure at the actuator
p_i	The i^{th} roadwheel penetration into the terrain
p_{ij}, p_{ji} p_{i1}, p_{j1} p_{i2}, p_{j2}	Points located on the i^{th} and j^{th} bodies respectively
P^u, P^v	Generalized impulses P partitioned according to u and v
q	Composite vector of all system generalized coordinates
Q	Composite vector of all system generalized forces
$\dot{q}(t_i-)$	System velocity before a discontinuous event at t_i
$\dot{q}(t_i+)$	System velocity after a discontinuous event at t_i
Q_A	Actuator flow rate
Q_C	Flow due to compressibility
q^i	Vector of generalized coordinates of body i
q^i, $\dot{q}^i$	Values of system generalized coordinates and velocities at t_i
Q^i	Vector of generalized forces on body i

Symbol	
$\Delta\dot{q}_i$	Discontinuous change in velocity at t_i
q_j, $\dot{q}_k$	The j^{th} element of q, the k^{th} element of $\dot{q}$
Q_L	Leakage flow
Q_p	Pump flow gradient
Q_T	Pump flow
Q^u, Q^v	Partitioning of Q according to u and v
Q^{v*}	Known generalized forces corresponding to the specified independent (v) generalized coordinates in a displacement driven system
Q^i_x, Q^i_y, Q^i_φ	Generalized forces on body i corresponding to generalized coordinates x_i, y_i, and φ_i, respectively
$Q^i_{x_s}$, $Q^i_{y_s}$, $Q^i_{\varphi_s}$	Generalized forces on body i due to a spring-damper-actuator combination
r	Number of independent algebraic constraint equations (rank of the constraint Jacobian matrix)
R(s)	Reference voltage signal from the control stick potentiometer
$\vec{R}_i$	Vector locating the origin of the coordinate system fixed in the i^{th} body
$\vec{r}_{ij}$, $\vec{r}_{ji}$	Vectors locating points on the i^{th} and j^{th} bodies, respectively, with respect to body fixed coordinate systems
$\vec{r}_{m_i}$	Vector locating the center of mass on the i^{th} body with respect to body fixed coordinate system
$\vec{r}_p$	Vector between two points on adjacent bodies
r_{sj}	Tracked vehicle sprocket radius
$\vec{r}_{s_{ij}}$, $\vec{r}_{s_{ji}}$	Vectors locating spring-damper-actuator attachment points on the i^{th} and j^{th} bodies, respectively, with respect to body fixed coordinate systems

Symbol	
$\vec{R}_{s_{ij}}$	Vector between spring-damper-actuator attachment points on two bodies
R^v	Unknown driving generalized forces corresponding to specified independent (v) generalized coordinates in a displacement driven system
r_w	Roadwheel radius
s	$s \equiv \dot{v}$ -- Introduced to reduce equations of motion to first order form
$\vec{S}_i$, $\vec{S}_j$, $\vec{r}_{ij}$	Vectors connecting points (p_{i1}, p_{i2}), (p_{j1}, p_{j2}), and (p_{j1}, p_{i1}) on the i^{th} and j^{th} body, respectively
s_{ij}, s_{ji}	Points located on the i^{th} and j^{th} bodies respectively
s^j	Time derivative of the j^{th} logical variable
Sj	Spring-damper-actuator numbers
S_j	Actual displacement of a reference point on the vehicle
S_{rj}	Total desired displacement of a reference point on the vehicle, along the terrain
S_{Oj}	Initial global location of a reference point on the vehicle
Slip	A number > 0 if the wheel slides in the positive x direction and < 0 if the wheel slides in the negative x direction
T^i	Kinetic energy of the i^{th} body
t_i^-	Approaches an event at t_i from $t < t_i$
t_i^+	Approaches an event at t_i from $t > t_i$
T_j	Track driving force developed by the power train
$T_L(s)$	Control stick limiting torque

Symbol	
T_q, $T_{\dot{q}}$	Partial derivative of kinetic energy with respect to generalized coordinates and velocities
u, v	Partitioning of q into dependent and independent variables, respectively
u*	Change in dependent generalized coordinates u due to time dependence of Φ
$\dot{u}$*	Change in dependent generalized velocities $\dot{u}$ due to time dependence of Φ
$\ddot{u}$*	Change in dependent generalized accelerations $\ddot{u}$ due to angular, centripetal, and Coriolis effects of constraints
u^i, v^i	Values of dependent and independent generalized coordinates at t_i
v_j	Actual velocity of a reference point on the vehicle
v^p, s^p	Predicted values of v and s
v_r	Desired velocity of a reference point on the vehicle
v^s, $\dot{v}^s$, $\ddot{v}^s$	Specified independent generalized coordinates, velocities, and accelerations in a displacement driven system
V_0	Entrained volume of hydraulic fluid
$v_{23}(s)$	Actuator velocity
w	$w \equiv \dot{q}$ -- Generalized velocity vector introduced to reduce equations of motion to first order
x, y	Inertial reference frame
x_B, y_B	Global location of the barrel
x_i, y_i	Global coordinates of the origin of the i^{th} body fixed coordinate system
x_{m_i}, y_{m_i}	Global coordinates of the i^{th} body center of mass
x_p, y_p	Global location of the pin in the weapon system

Symbol	
x_{p_j}, y_{p_j}	Global location of a point on the j^{th} body
x_R, y_R	Global location of the receiver in the weapon system
$Y(s)$	Pump yoke displacement
$z(s)$	Actuator displacement
α	Angle between a spring-damper-actuator and the inertial reference frame
α	$\alpha = h(x)$
α_j, β_0	Gear's coefficients
β_e	Effective bulk modulus including system compliance
δ	Force unbalance tolerance
δW^i	Virtual work of forces acting on the i^{th} body
ϵ	Constraint closure tolerance
$\theta_R(s)$	Angular displacement of control stick
λ	Lagrange multiplier vector that determines constraint reaction forces
$\tilde{\lambda}$	Vector of Lagrange multipliers that determine constraint reaction impulses
λ^1, λ^2	Lagrange multipliers corresponding to independent and dependent constraint equations
μ	Coefficient of friction
ξ_i, η_i	Body-fixed coordinate systems in the i^{th} body
ξ_{ij}, η_{ij}	Components of the vectors, $\vec{r}_{ij}$, $\vec{r}_{i1}$, $\vec{r}_{i2}$, and $\vec{r}_{s_{ij}}$, respectively, in the ξ_i - η_i coordinate system
ξ_{i1}, η_{i1}	
ξ_{i2}, η_{i2}	
$\xi_{s_{ij}}$, $\eta_{s_{ij}}$	

ξ_{ji}, η_{ji} ξ_{j1}, η_{j1} ξ_{j2}, η_{j2} $\xi_{s_{ji}}$, $\eta_{s_{ji}}$	Components of the vectors $\vec{r}_{ji}$, $\vec{r}_{j1}$, $\vec{r}_{j2}$, and $\vec{r}_{s_{ji}}$, respectively, in the $\xi_j - \eta_j$ coordinate system
ξ_{m_i}, η_{m_i}	Components of the vector $\vec{r}_{m_i}$ in the $\xi_i - \eta_i$ coordinate system
τ_c	Control system time constant
τ_1	Servo valve time constant
τ_1, τ_2	Small time interval containing a discontinuous event
Φ	Vector of equations of constraint between bodies
φ_i	Angle between the inertial reference x-axis and body-fixed ξ_i-axis
Φ_q, $\partial\Phi/\partial q$ $[\partial\Phi_i/\partial q_j]$	Constraint Jacobian matrix
$\bar{\Phi}_q$	Nonsingular matrix composed of Φ_q and $\bar{I}$
Φ_t, Φ_{tq}, Φ_{tt}	Partial derivatives of constraints with respect to time and generalized coordinates
Φ_u, Φ_v	Partial derivatives of constraints with respect to dependent and independent variables, respectively
Φ^1	Independent constraint equations
Φ^2	Dependent or redundant constraint equations
Ω^j	The j^{th} logical function
cond( )	Condition number of a matrix
δ( )	A virtual change in the indicated quantity
Δ( )	A small change in the indicated quantity

Symbol	
$\dot{(\ )}, \ddot{(\ )}$	Differentiation with respect to time
$\|\| \cdot \|\|$	The ℓ^2 norm of a vector
$(\ )^T$	Transpose of a vector or matrix
$(\ )^{-T}$	Inverse of the transpose of a matrix
$(\ )^{-1}$	Inverse of a matrix

CHAPTER 1

INTRODUCTION

1.1 Introductory Comments

Analysis of machine and mechanism problems prior to 1960 relied upon various (and often quite complex) graphical techniques. These techniques were based on geometrical interpretation of constraints and a large body of literature evolved which classified constraints and linkages [1, 2]. In the 1950's digital and analog computers began to appear, making it possible to formulate and repetitively solve moderately sized systems of equations [3, 4]. This prompted the development of mathematical software support packages and the possibility of mathematical analysis of machines and vehicular systems became a reality. The early programs, written specifically for given applications, used geometrical interpretation of constraints to define relative position coordinates, thus helping to reduce problem size. These programs were generally limited in scope, requiring extensive modification for different applications [5, 6].

As computer speed and capacity increased, larger programs were developed, along with data handling routines. It was recognized that mechanical systems are composed of a number of "standard" elements and that they can be combined as building blocks to define

large classes of mechanisms. Again, geometrical interpretation of constraints allowed for the reduction in the number of independent variables, so that considerably larger, more sophisticated models could be established. However, the methods developed required topological analysis associated with identification of independent constraint loops, which added to program complexity.

Several general purpose computer programs were developed along these lines in the late 60's and early 70's [7-9]. These programs are satisfactory for many mechanisms applications, but incorporation of user supplied constraint and forcing functions is rather difficult.

An alternate method of formulating system constraints and equations of motion, in terms of a maximal set of coordinates, bypasses topological analysis and provides for convenient user supplied constraints and forcing functions [10]. This leads to a more general computer program, with practically no limitation on mechanism type. The penalty, however, is a larger system of equations to be solved, which may require greater computer capacity or time.

1.2 Program Efficiency Versus Program Generality

The first general purpose computer codes capable of performing dynamic analysis of large scale, nonlinear, constrained, rigid body mechanical systems appeared around 1970. The earliest programs having major impact in this area were DYMAC (1970) [7], DAMN-DRAM (1971) [8], IMP (1972) [9], and ADAMS-3D (1973) [10]. Numerous programs have since been developed but they generally represent variations on the above algorithms [11-14]. These programs are

similar in some respects and are quite different in others. Table 1.1 lists some of the major features of the above four programs.

In the design of algorithms it is intended that program efficiency and generality be maximized. Efficiency is easy to measure. Generality, however, is highly dependent upon the needs of the user and a program that is ideally suited for certain applications may be completely inadequate for others. Therefore it is difficult to assign ratings to generality of a given code.

Many codes are developed using linkages as subelements of mechanical systems. Since linkages generally form loops (closed or open) a minimum number of linkage relative position coordinates (designated as Lagrangian coordinates by Paul [15]) are defined, allowing loop constraint equations to be formulated. Complex systems may have many loops (not all independent) so graph theory or user preprocessing is required to avoid unnecessary formulation of redundant loop constraint equations [16, 17]. Loop constraint formulation and topological preprocessing requirements add an order of magnitude of complexity to codes when the user desires to incorporate constraint relations that are not provided in the standard formulation. A definition of program generality might then be a measure of how easily one supplements the standard code to provide for unanticipated constraint or forcing elements.

Constraint loop formulations, with topological preprocessing, are desirable from a numerical standpoint since they allow formulation of a problem using a minimum number of state variables. They

Table 1.1. Comparison of features of some general purpose computer codes

	DYMAC	DRAM	IMP	ADAMS	DADS
Model Dimension	2D	2D	3D	2D/3D	2D/3D
Systems Modeled	Closed Loop	Closed/Open Loop	Closed Loop	Closed/Open Loop	Closed/Open Loop
Coordinate Type	Lagrangian	Lagrangian	Lagrangian	Cartesian	Cartesian
Loop Constraint/Graph Theory	Yes	Yes	Yes	No	No
User Constraints/ Forcing Functions	Moderately Difficult	Moderately Difficult	Moderately Difficult	Moderately Easy	Easy
Lagrange Multipliers	No	Yes	No	Yes	Yes
Identification of Independent Variables	User	No	Program	No	Program
Integrate for Independent Variables Only	Yes	No	Yes	No	Yes
Sparse Matrix Techniques	No	No	No	Yes	Yes/No

lead to a corresponding minimum number of highly coupled constraint equations. The constraint Jacobian matrix resulting from linearization of constraint equations will generally be small, even for relatively large problems. Well established full matrix manipulation algorithms can be efficiently applied. The small number of state variables thus yields a small number of differential equations of motion, which is a definite advantage for numerical integration algorithms [18].

The programs DYMAC, DRAM, and IMP employ loop methods, as described above. A different approach, however, is taken in ADAMS [10, 19] whereby constraint equations are formulated between bodies connected by given joints. The concept of link is not used in this development. Elements of the system are treated as rigid bodies each with a Cartesian coordinate system attached. One does not think in terms of element type, i.e., slider, crank, rocker, etc.; but in terms of joint type, i.e., revolute, translational, cylindrical, etc. Since each system element has an identical coordinate system, arbitrary user-supplied constraints and generalized forcing functions can be formulated without regard to element type. Thus, equation formulation and computer programming are simplified.

A major disadvantage of this method arises from initially assigning a maximal set of generalized coordinates (degrees of freedom) to the system. Nonlinear algebraic constraint equations are then imposed to remove system degrees of freedom. This results in a maximal number of differential and algebraic equations, which

traditionally have been solved iteratively and simultaneously by stiffly stable implicit numerical integration algorithms [20]. Solving large systems of equations iteratively, even when advantage is taken of matrix sparsity, is time consuming. In addition, the corrector equations thus obtained contain variables related to accelerations, on which error control cannot be maintained. Poor prediction of these variables results in an excessive number of corrector iterations (up to five or more), forcing even smaller time steps than would otherwise be necessary.

In summary, the loop method of constraint formulation gains program efficiency at the expe se of program generality, whereas the joint method of constraint formulation in ADAMS gains program generality at the expense of program efficiency. The objective of the research reported in this report is development of a numerical analysis method that contains the strong points of both these methods.

1.3 Obtaining Efficiency Without Sacrificing Generality

The main objective of this research is to develop numerical and analytical methods to improve the efficiency of general purpose analysis programs that determine dynamic response of large scale, constrained, rigid-body mechanical systems. A planar dynamic analysis code [10, 19, 21], similar to ADAMS, formulates nonlinear holonomic constraint equations and differential equations of motion, written in terms of a maximal set of Cartesian generalized coordinates, to facilitate the general formulation of constraints and

forcing functions. A maximal set of coupled algebraic and differential equations are then solved iteratively and simultaneously by an implicit numerical integration algorithm [20]. As noted earlier the method is inefficient (even though sparse matrix manipulation algorithms are employed), because an excessive number of equations are involved in the iteration process within the numerical integration algorithm.

A modified approach developed in this report maintains program generality by keeping the above method of constraint and equation of motion formulation. An intermediate numerical processing step is introduced that effectively eliminates the constraint equations and dependent equations of motion, before each numerical integration step. Iteration is then limited to solving only the set of constraint equations. The time saved in the iteration step and in the numerical integration step far outweighs the overhead introduced by the intermediate numerical processing step. This method has demonstrated an order of magnitude increase in speed, when highly constrained systems are simulated.

A Gaussian elimination algorithm with full pivoting [22] decomposes the constraint Jacobian matrix and identifies dependent and independent generalized coordinates. For small systems, the algorithm constructs an influence coefficient matrix relating variations in dependent and independent variables. For large systems, in which matrix sparsity becomes a significant factor, the algorithm provides information necessary to set up a modified sparse matrix

that relates variations in dependent and independent variables. In either case, this information is employed to numerically construct a reduced system of differential equations of motion, whose solution yields the total system dynamic response.

A pieced interval analysis concept is also developed to further the enhancement of program efficiency. When rapidly changing or "intermittent" events occur within a simulation interval, it may be desirable or necessary to break the time interval into subintervals. Different equations or analysis techniques may then be incorporated at these subintervals. For example, numerical integration algorithms employing high order polynomial predictors become inefficient or fail near abrupt or discontinuous variations of state variables. When event times are known in advance, time stepsize control can be placed on the integration algorithm before encounter, to avoid lengthy search for correct reduced time step sizes. Momentum balance is performed to determine jump discontinuity in velocity if bodies impact or impulsive loading occurs.

Event occurrences, such as impact between bodies, are generally functions of system state or time. Therefore an event predictor is incorporated into the integration algorithm. Logical functions of state and time are formulated such that, as each given function passes through zero, it defines a logical time at which other actions may be taken. Logical functions are extrapolated ahead before the system state is advanced, so that significant events may

be detected before being encountered by the system. The logical time corresponding to a zero of a logical function is determined and the system solution is forced precisely at this time.

1.4 Scope of the Report

Both planar and three dimensional rigid-body dynamic analysis programs, employing Cartesian coordinates and a simple constraint formulation, have been developed [10, 19, 21, 23, 24]. In addition, flexible beam linkage elements [25] have been incorporated into the two dimensional program to extend its range of applications. The analysis methods developed in this research apply equally well to each of the above programs. In the interest of economy only the planar rigid-body analysis method is developed here. This is sufficient to illustrate the analysis procedure, without introducing unnecessary complexity.

In Chapter 2, the planar rigid-body dynamic analysis algorithm is developed. An implicit, stiffly stable numerical integration algorithm is employed to simultaneously and iteratively solve algebraic constraint equations and differential equations of motion. In addition, it is shown that this method of solution results in a sparse Jacobian matrix of maximal dimension, and that sparse matrix manipulation algorithms must be employed to gain efficiency, even for small problems. Further, it is shown that the method requires an excessive number of corrector iterations, employing the above matrix at each time step.

In Chapter 3 the "inverse dynamics problem" is presented [26]. That is, assuming that motion of certain generalized coordinates (equal in number to system degree of freedom) is specified, the entire system state, including all external, inertial, and constraint forces can be determined. This is done by an iterative algebraic procedure, requiring no numerical integration. Methods for reducing problem size by numerical elimination of constraint equations and dependent variables are more easily demonstrated for the inverse dynamics problem. However, these methods are equally applicable to the dynamics problem of Chapter 4.

It is shown that the Jacobian matrix formed by linearizing the constraint equations expresses relations between variations in the generalized coordinates. The above matrix also expresses similar relations between higher order derivatives. A Gaussian elimination procedure is employed to identify a submatrix of maximal rank and it is shown, by the implicit function theorem, that generalized coordinates corresponding to certain columns of this matrix depend entirely on the remaining coordinates and possibly time, hence they are designated dependent variables. These dependent variables may be evaluated if the independent variables are known, since this submatrix is nonsingular. Likewise, dependent velocities and accelerations are evaluated, when corresponding independent velocities and accelerations are known. Constraint reaction forces, included in the equations of motion, are given by the product of the transpose of the above matrix and a vector of scaler multipliers. Eliminating

the scaler multipliers, employing nonsingularity of the above submatrix, yields a minimal set of equations of motion, involving only independent variables and excluding all constraint reaction forces.

A significant number of intermediate matrix products are involved in the above procedure. For large systems, efficiency would be lost when carrying out the indicated sparse matrix products. Therefore, an alternate sparse matrix method is developed that requires no intermediate matrix products, yet effectively arrives at the same minimal set of equations of motion.

An additional problem arises when solving nonlinear equations by an iterative procedure, such as Newton's method. If the initial estimate of roots of a system of nonlinear equations is poor, convergence will be slow or may fail. While the methods developed in Chapter 3 minimize the effect of errors in dependent variables (on convergence rate), better estimates will result in fewer or quite often no iterations. When advancing a constrained system from one position to another, if dependent velocities and accelerations are determined, they may be used to arrive at better extrapolated values for dependent position variables. Alternately, a variable order polynomial extrapolator fitted to dependent variable history will yield even better results. Newton's backward divided difference formula is very convenient and efficient for this application [27].

The numerical procedures of Chapter 3 are extended to the direct dynamics problem in Chapter 4 [26]. Here it is assumed that one or more of the system generalized coordinates are not given

as functions of time. Therefore, the equations of motion must be solved by numerical integration to determine system response to forces. The minimal system of differential equations of motion may then be conveniently solved by explicit predictor/implicit corrector multistep methods, such as the Adams Bashforth/Adams Moulton method [28].

Additional problems arise as a result of identifying optimal sets of independent variables. The sets generally change as system configuration changes. Therefore, the minimal set of differential equations also changes. To allow for the possibility that any variable may become independent, polynomial extrapolators are maintained for all dependent variables. A given polynomial extrapolator automatically changes to a predictor if the corresponding variable becomes independent, and a predictor reverts back to an extrapolator when a variable becomes dependent. This procedure provides accurate estimates for the Newton iteration described earlier for solution of constraint equations and avoids interruption of the numerical integration procedure caused by lack of variable history.

The concept of pieced interval analysis is introduced in Chapter 5. In the interest of efficiency and simplicity a given simulation may be divided into two or more time intervals for which different governing equations or analysis procedures may apply. Boundaries of such intervals may correspond to instances where violent actions such as member impact, impulsive loading, mass capture, mass release, member breaking, etc. occurs. Such events,

when treated in a continuous manner, create serious problems for high order predictor/corrector integration algorithms, when their presence is not known in advance [29]. The ability to force small step sizes and/or relax error requirements just prior to event encounter is an effective means of improving program efficiency. If member distortion is significant, a temporary flexible model may be implemented. Times at which such events occur are usually not known in advance, since they depend on system state. Therefore, logical functions of system state and time, supplied by the user, are employed by the integration algorithm to predict event occurrences before encounter. These logical event occurrences define logical times that mark the interval boundaries.

A logical events monitor is developed in Chapter 5. In addition, momentum balance equations for general planar constrained rigid body systems undergoing impulsive loading or impact are derived. Incorporation of flexible degrees of freedom into the model has been dealt with [25], and will not be included in this report. It is interesting to note that if a flexible model is required on an interval for which small angular rotations occur, the problem is essentially linear and efficient modal analysis techniques may be employed.

Numerical examples are presented in Chapter 6 to demonstrate efficiency and generality of the method. However, a detailed description of the operational and data requirements of the computer program is not presented in this report. This information will be available

in a theoretical and user's manual that will allow periodic update, since much of the program is still in a developmental stage. Examples are presented in enough detail to demonstrate the contributions of this research. In addition, they suggest applications not found in general purpose mechanical systems analysis programs.

The first example treated investigates dynamic pitch response of an articulated vehicle, consisting of two tracked-vehicles that are coupled together by an articulation joint and a hydraulic actuator. A representative model of the mechanical vehicular system is developed, primarily from the standard program element formulation. Additional user supplied generalized forcing functions and constraints are formulated to incorporate major contributions to system dynamic response. An electro-mechanical-hydraulic control system, with position and force feedback, is formulated and coupled to the mechanical system through extensions of a standard element in the program. This system controls relative vehicular pitch attitude by monitoring hydraulic actuator extension rate and system pressure. The control system differential equations and mechanical system differential equations of motion are solved simultaneously by the same integration algorithm. Thus, no iterative procedures or approximations are required to obtain the coupled system dynamic response.

A second example illustrates the pieced interval analysis capability of the program. Recoiling dynamics of a weapon system, subject to discontinuous and impulsive forces, impacts, and mass

capture and release is investigated. In addition, the system contains a cammed loading mechanism that affects barrel dynamics. Logical functions are developed for this intermittent motion problem. It is shown how various programmed actions are taken, in response to each active logical function. For this problem, three actions in order of complexity are:

1. Remove or add forces to the system. Restart the integration process.
2. Apply impulsive loads to the system. Perform momentum balance to determine jump discontinuities in velocity. Restart the integration process.
3. Impact, mass capture, or mass release occurs. Introduce restitution equations and perform momentum balance to determine jump discontinuities in velocity. Impose or remove constraints for mass capture or release, respectively. Restart the integration process.

CHAPTER 2

THE PLANAR RIGID BODY MODELING METHOD FOR CONSTRAINED SYSTEMS

2.1 Introduction

The term "Rigid Body Systems" distinguishes the intent of the analysis method and program developed in this chapter. Many of the analysis programs existing today are based on the traditional notion of "machines and mechanisms," in which the ideas of "links" and "closed loop analysis" are embedded in the "geometry of mechanisms". This limitation complicates an orderly approach to general constrained dynamic system analysis.

Lagrange may have been the first to recognize the limitation of the geometrical approach, as evidenced by the statement found in the preface of his Mecanique Analytique [30]: "No diagrams will be found in this work. The methods that I explain in it require neither constructions nor geometrical or mechanical arguments, but only the algebraic operations inherent to a regular uniform process. Those who love Analysis will, with joy, see mechanics become a new branch of it and will be grateful to me for thus having extended its field."

The concept of constraints in physical systems is not unlike the concept of circuits in electrical systems. Kirchhoff stated two basic laws that: (1 - voltage law) "The voltages with their

proper signs, taken completely around a mesh, add up to zero", and (2 - current law) "The currents at a node, with due regard to direction, add up to zero." From these laws and Ohm's law, two basic methods of circuit solution have developed as follows:

(a) the mesh method of solution, characterized by the summation of voltages around meshes, is an application of the voltage law.

(b) the nodal method, characterized by the summation of currents at nodes, is an application of the current law.

Loop-closure methods of mechanism analysis are analogous to the mesh method, (a). An alternate and more direct approach, first developed on a large scale by Orlandea [10] for general three dimensional analysis, is based on the nodal method, (b). He called this method "the Node - Analogous Formulation" for mechanical systems, derived by considering that D'Alembert's Principle for forces and Kirchhoff's law for currents are analogous. He further identified a finite number of standard components of mechanical systems (linkages). While his method easily allows all of the constraints associated with linkages, there need be no connotation of linkages in its development.

The analysis method presented here is developed according to the nodal representation, as introduced by Orlandea. Since this method is nearly devoid of geometrical and topological concepts, it is adapted to a broad class of mechanical systems. These systems may range from the most complex linkage mechanisms to articulated tracked vehicles with electro-hydraulic control systems.

In this chapter, a general system of constrained equations of rigid body planar motion is formulated. An implicit numerical integration method for automatically solving the system equations and sparse matrix techniques are also discussed. Generalized coordinates are defined to locate each individual rigid member of the system and to express kinetic energy for each member. Constraints between elements are taken as friction-free standard constraints, with provisions for additional, non-standard constraint formulation, as needed. In addition to standard constraints, springs, dampers, and actuators connecting any pair of points on different bodies of the system are included in the model. These standard force elements, together with allowance for arbitrary non-standard forcing functions, make the formulation quite general.

Implicit numerical integration and sparse matrix algorithms [31, 32] were initially used in numerical integration of the equations of motion. After considerable numerical experimentation, it became apparent that implicit simultaneous solution of algebraic and differential equations create artificially stiff [20] problems and the integration of large systems of equations increases the spectrum of frequencies with which integration algorithms must cope. This prompted a search for new methods of solving the system equations of motion, which forms the basis of the method presented in the following chapters. The implicit method is presented in this chapter, for comparison purposes only. The basic constraint

formulation and equations of motion presented in this chapter are used as the basis for developments in the following chapters.

2.2 Constrained Equations of Planar Motion

(a) Generalized Coordinates and Energy Equations

In order to determine the configuration or state of a planar mechanical system, it is first necessary to define generalized coordinates that specify the location of each body in the system. As shown in Fig. 2.1, let the x-y coordinate system be a fixed inertial reference frame. Define a body-fixed $\xi_i - \eta_i$ coordinate system embedded in a typical body i. The location of body i is specified by the global coordinates (x_i, y_i) or the vector $\vec{R}_i$ of the origin of its reference frame and the angle φ_i of rotation of the body fixed ξ_i-axis, relative to the global x-axis.

The center of mass of body i is located by a vector $\vec{r}_{m_i}$ in the body fixed coordinate system, with $\xi_i - \eta_i$ components ξ_{m_i} and η_{m_i}. In terms of the generalized coordinates x_i, y_i, and φ_i and the parameters ξ_{m_i} and η_{m_i}, the center of mass is located globally by

$$x_{m_i} = x_i + \xi_{m_i} \cos \varphi_i - \eta_{m_i} \sin \varphi_i$$

$$y_{m_i} = y_i + \xi_{m_i} \sin \varphi_i + \eta_{m_i} \cos \varphi_i \qquad (2.1)$$

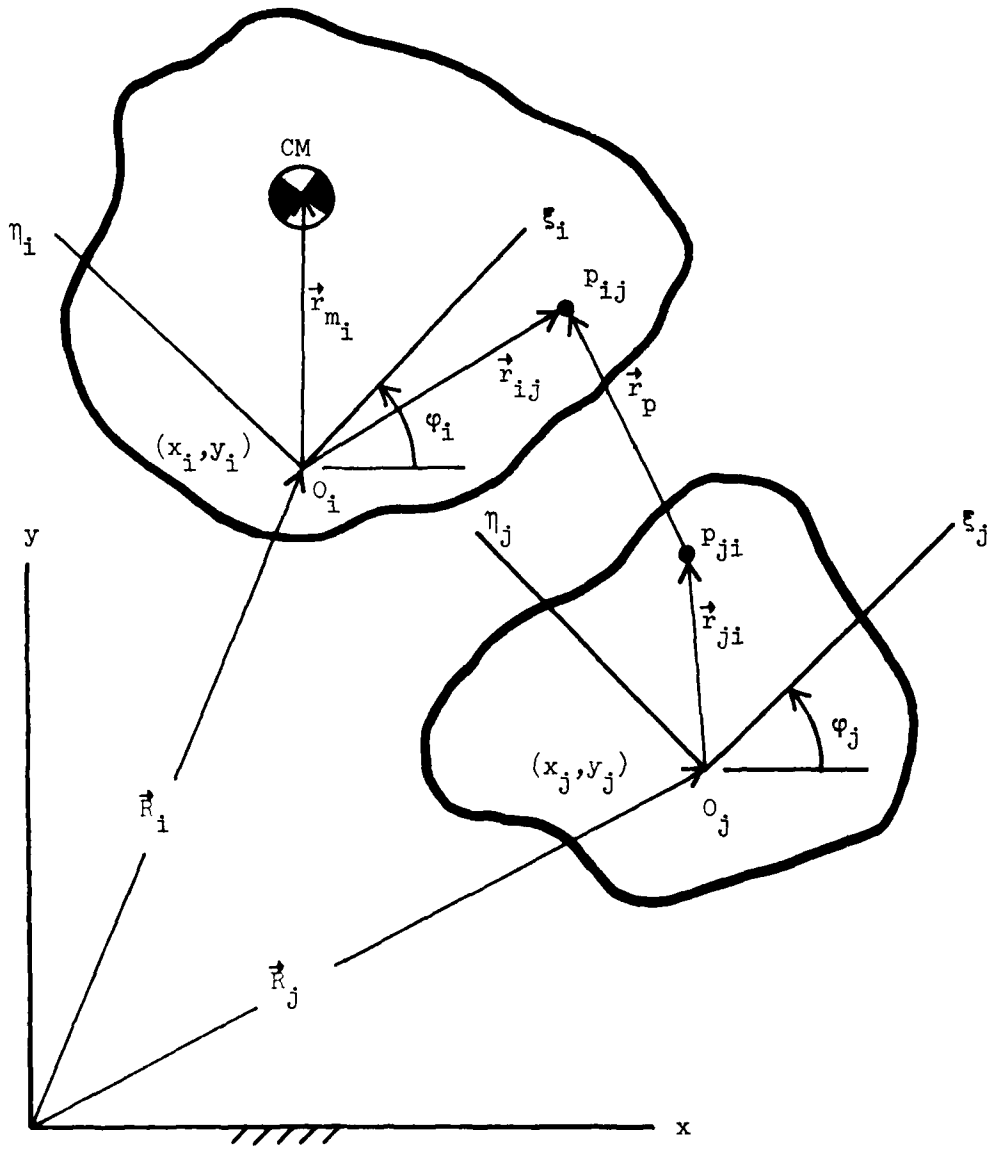


Figure 2.1. Rigid bodies connected by a vector $\vec{r}_p$ and rotational joints.

Thus, the kinetic energy of body i is

$$T^i = \frac{1}{2} m_i \left(\dot{x}^2_{m_i} + \dot{y}^2_{m_i} \right) + \frac{1}{2} J_i \dot{\varphi}_i^2 \qquad (2.2)$$

where m_i is the mass of body i and J_i is its centroidal polar moment of inertia.

(b) Equations of Constraint

Figure 2.1 further depicts an adjacent body j, with body-fixed coordinate system located by the vector $\vec{R}_j$. Let arbitrary points p_{ij} on body i and p_{ji} on body j be located by vectors $\vec{r}_{ij}$ and $\vec{r}_{ji}$, specified in the body fixed coordinate systems by coordinates ξ_{ij}, η_{ij}, ξ_{ji}, and η_{ji}. These points are in turn connected by a vector $\vec{r}_p$,

$$\vec{r}_p = \vec{R}_i + \vec{r}_{ij} - \vec{R}_j - \vec{r}_{ji} \qquad (2.3)$$

The vector condition for a rotational joint between bodies i and j, at points p_{ij} and p_{ji}, is simply $\vec{r}_p = \vec{0}$, yielding the following pair of scaler constraint equations:

$$x_i + \xi_{ij} \cos \varphi_i - \eta_{ij} \sin \varphi_i - x_j - \xi_{ji} \cos \varphi_j$$

$$+ \eta_{ji} \sin \varphi_j = 0$$

$$y_i + \xi_{ij} \sin \varphi_i + \eta_{ij} \cos \varphi_i - y_j - \xi_{ji} \sin \varphi_j$$

$$- \eta_{ji} \cos \varphi_j = 0 \qquad (2.4)$$

If $\vec{r}_p$ in Fig. 2.1 is taken as a nonzero vector of fixed length r_p, a "massless link" of length r_p connects points p_{ij} and p_{ji}. A single scalar equation for this constraint can then be written as

$$\| \vec{r}_p \|^2 - r_p^2 = (x_i + \xi_{ij} \cos \varphi_i - \eta_{ij} \sin \varphi_i - x_j - \xi_{ji} \cos \varphi_j$$

$$+ \eta_{ji} \sin \varphi_j)^2 + (y_i + \xi_{ij} \sin \varphi_i + \eta_{ij} \cos \varphi_i$$

$$- y_j - \xi_{ji} \sin \varphi_j - \eta_{ji} \cos \varphi_j)^2 - r_p^2 = 0 \qquad (2.5)$$

For a translational joint shown in Fig. 2.2, let points p_{i1} and p_{i2} on body i, and points p_{j1} and p_{j2} on body j lie on some line parallel to the path of relative motion between the two bodies, such that the specified body fixed vectors $\vec{S}_i$ and $\vec{S}_j$ are of nonzero magnitude. Since $\vec{S}_i$ and $\vec{S}_j$ are parallel, $\vec{S}_i \times \vec{S}_j = \vec{0}$, with zero z component yielding the scalar equation

$$[(\xi_{i2} - \xi_{i1}) \cos \varphi_i - (\eta_{i2} - \eta_{i1}) \sin \varphi_i]$$

$$[(\xi_{j2} - \xi_{j1}) \sin \varphi_j + (\eta_{j2} - \eta_{j1}) \cos \varphi_j]$$

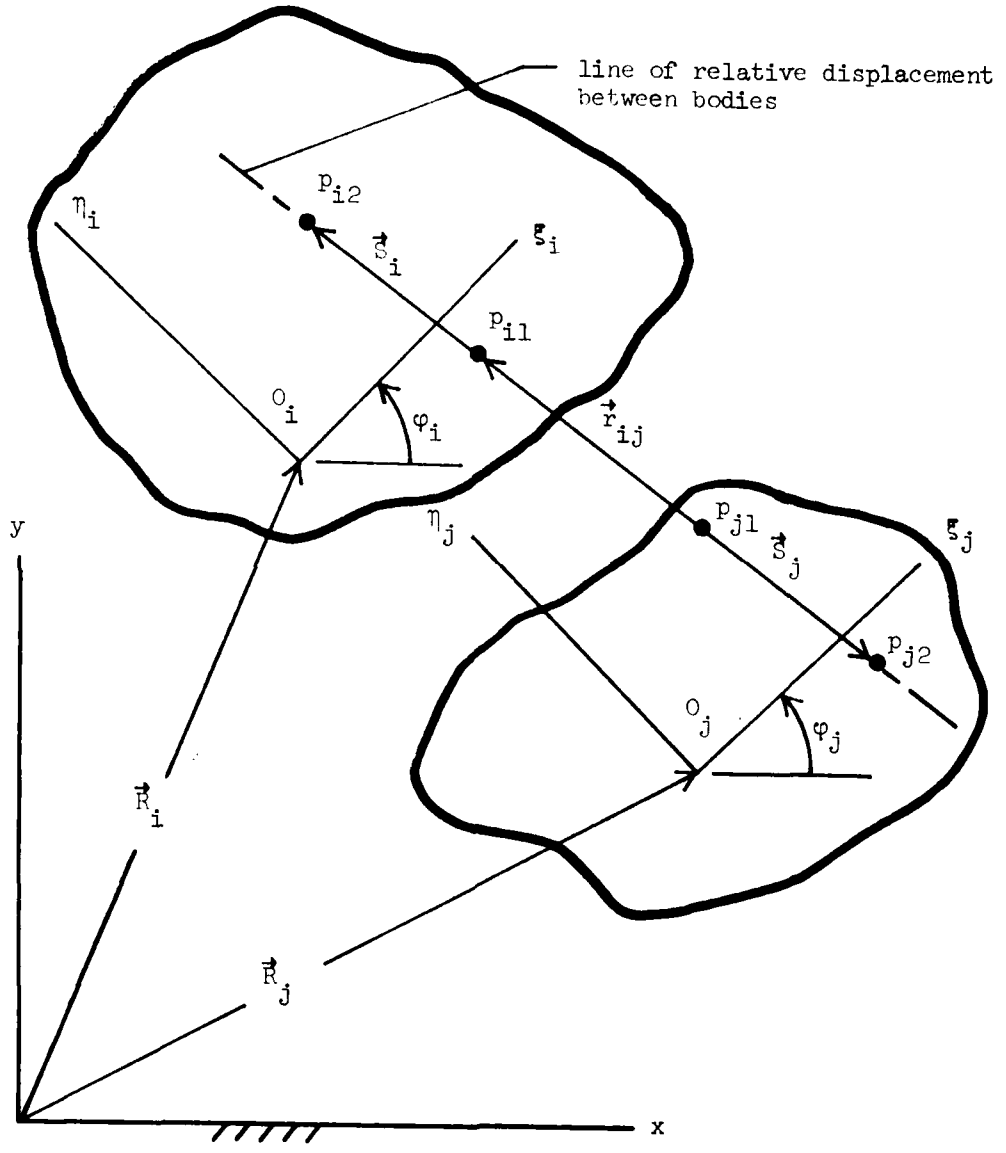


Figure 2.2. Rigid bodies connected by a translational joint.

$$- [(\xi_{j2} - \xi_{j1}) \cos \varphi_j - (\eta_{j2} - \eta_{j1}) \sin \varphi_j]$$

$$[(\xi_{i2} - \xi_{i1}) \sin \varphi_i + (\eta_{i2} - \eta_{i1}) \cos \varphi_i] = 0 \qquad (2.6)$$

Likewise, $\vec{r}_{ji}$ and $\vec{S}_j$ are parallel, so $\vec{r}_{ji} \times \vec{S}_j = \vec{0}$ yields a zero z component and the second scalar equation

$$[x_i + \xi_{i1} \cos \varphi_i - \eta_{i1} \sin \varphi_i - x_j - \xi_{j1} \cos \varphi_j + \eta_{j1} \sin \varphi_j]$$

$$[(\xi_{j2} - \xi_{j1}) \sin \varphi_j + (\eta_{j2} - \eta_{j1}) \cos \varphi_j]$$

$$- [y_i + \xi_{i1} \sin \varphi_i + \eta_{i1} \cos \varphi_i - y_j - \xi_{j1} \sin \varphi_j - \eta_{j1} \cos \varphi_j]$$

$$[(\xi_{j2} - \xi_{j1}) \cos \varphi_j - (\eta_{j2} - \eta_{j1}) \sin \varphi_j] = 0 \qquad (2.7)$$

The parameters (ξ_{i1}, η_{i1}) and (ξ_{i2}, η_{i2}) locate points p_{i1} and p_{i2} in body i coordinate system, and (ξ_{j1}, η_{j1}) and (ξ_{j2}, η_{j2}) locate points p_{j1} and p_{j2} in body j coordinate system.

Other constraints may be formulated by a similar process. Denote by $q^i = [x_i, y_i, \varphi_i]^T$ the vector of generalized coordinates of body i and by $q = [q^{1^T}, q^{2^T}, \ldots, q^{n^T}]^T$ the composite vector of all system generalized coordinates. In this notation, the holonomic constraints of Equations 2.4 to 2.7 and other (perhaps time dependent) holonomic constraints can be written in vector function form as

$$\Phi(q,t) = 0 \tag{2.8}$$

where $\Phi(q,t) = [\Phi_1(q,t), \ldots, \Phi_m(q,t)]^T$ and the functions Φ_i, $i = 1, \ldots, m$, are assumed to be independent, i.e., the Jacobian $\partial\Phi/\partial q \equiv [\partial\Phi_i/\partial q_j]$ has full row rank.

(c) Equations of Springs, Dampers and Actuators

Internal forces due to other types of elements acting between bodies may be obtained by a process similar to the constraint equation development. For example, since springs, dampers, and actuators shown in Fig. 2.3 generally appear together, they are incorporated into a single set of equations. The equation for spring-damper-actuator force is

$$\vec{F}_{ij} = \left[k_{ij}(\ell_{ij} - \ell_{O_{ij}}) + c_{ij}\,\dot{\ell}_{ij} + F_{O_{ij}} \right] \frac{1}{\ell_{ij}} \vec{R}_{s_{ij}} \tag{2.9}$$

where $\vec{F}_{ij}$ is the resultant force vector $F_{x_{ij}}\,\vec{i} + F_{y_{ij}}\,\vec{j}$ in the element, $\vec{R}_{s_{ij}}$ is the vector $\ell_{ij} \cos\alpha\,\vec{i} + \ell_{ij} \sin\alpha\,\vec{j}$ between points s_{ij} and s_{ji}, k_{ij} is an elastic spring coefficient that may depend on generalized coordinates and time, c_{ij} is a damping coefficient that may depend on generalized coordinates and time, $\ell_{O_{ij}}$ is the undeformed spring length, ℓ_{ij} is the deformed spring length, $\dot{\ell}_{ij}$ is the time derivative of ℓ_{ij}, and $F_{O_{ij}}$ is an actuator force applied along the element that may depend upon generalized coordinates and time. The unit vectors $\vec{i}$ and $\vec{j}$ are parallel to the x and y axes, respectively.

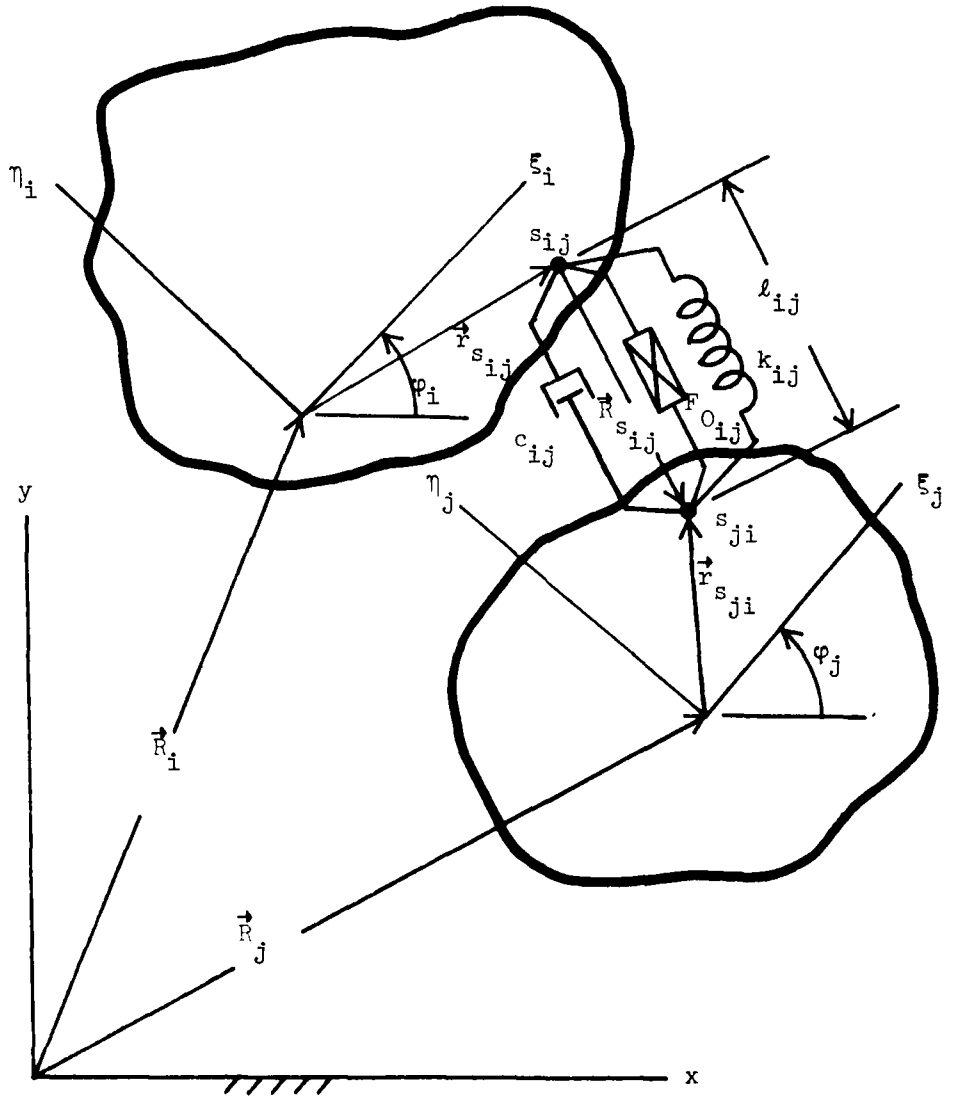


Figure 2.3. Rigid bodies connected by a spring-damper-actuator combination.

(d) Conservative and Nonconservative Generalized Forces

Contributions of forces acting on body i to the system equations of motion are determined by employing a virtual work concept. The virtual work of externally applied forces and spring-damper-actuator forces acting on body i is written as

$$\delta W^i = Q_x^i(q,\dot{q},t)\ \delta x_i + Q_y^i(q,\dot{q},t)\ \delta y_i + Q_\varphi^i(q,\dot{q},t)\ \delta\varphi_i \qquad (2.10)$$

The vector $Q^i = [Q_x^i, Q_y^i, Q_\varphi^i]^T$ of generalized forces on body i is thus defined and $Q = [Q^{1^T}, Q^{2^T}, \ldots, Q^{n^T}]^T$ is the vector of system generalized forces. Typical generalized forces are calculated to illustrate the procedure. Consider here the spring-damper-actuator element of Fig. 2.3 where point s_{ij} on body i is located by the vector $\vec{R}_i + \vec{r}_{s_{ij}}$. A virtual change in the location of point s_{ij} is given by

$$\delta\left(\vec{R}_i + \vec{r}_{s_{ij}}\right) = \frac{\partial}{\partial x_i}\left(\vec{R}_i + \vec{r}_{s_{ij}}\right)\delta x_i + \frac{\partial}{\partial y_i}\left(\vec{R}_i + \vec{r}_{s_{ij}}\right)\delta y_i$$

$$+ \frac{\partial}{\partial\varphi_i}\left(\vec{R}_i + \vec{r}_{s_{ij}}\right)\delta\varphi_i \qquad (2.11)$$

Forming the dot product of $\vec{F}_{ij}$ with $\delta\left(\vec{R}_i + \vec{r}_{s_{ij}}\right)$ determines contributions to the generalized force expressions for body i as

$$Q^i_{x_s} = F_{x_{ij}}$$

$$Q^i_{y_s} = F_{y_{ij}}$$

$$Q^i_{\varphi_s} = F_{x_{ij}} \left(- \xi_{s_{ij}} \sin \varphi_i - \eta_{s_{ij}} \cos \varphi_i \right)$$

$$+ F_{y_{ij}} \left(\xi_{s_{ij}} \cos \varphi_i - \eta_{s_{ij}} \sin \varphi_i \right) \qquad (2.12)$$

Generalized force contributions are summed for all externally and internally applied forces acting on body i, to arrive at Q^i.

(e) System Equations of Motion

Virtual displacements δq that are consistent with constraints (i.e., with time fixed) satisfy

$$\Phi_q \, \delta q = 0 \qquad (2.13)$$

where $\Phi_q = \partial \Phi / \partial q \equiv [\partial \Phi_i / \partial q_j]_{m \times 3n}$ using subscript notation for differentiation with respect to a vector.

The variational form of Lagrange's equations of motion, with workless constraints, is [33]

$$[\frac{d}{dt} (T_{\dot{q}}) - T_q - Q^T] \, \delta q = 0 \qquad (2.14)$$

which must hold for all δq satisfying Equation 2.13. By Farkas' Lemma [34], there exists a vector $\lambda \in R^m$ of multipliers (called Lagrange multipliers) such that

$$\frac{d}{dt}\left(T_{\dot{q}}\right)^T - \left(T_q\right)^T - Q - \Phi_q^T \lambda = 0 \tag{2.15}$$

which with Equation 2.8 form the constrained equations of motion of the system. For planar systems treated here, Equations 2.1 and 2.2 yield

$$\frac{d}{dt}\left(T^i_{\dot{q}^i}\right)^T - \left(T^i_{q^i}\right)^T = M^i\left(q^i\right)\ddot{q}^i - A^i\left(q^i, \dot{q}^i\right) \tag{2.16}$$

where

$$M^i = \begin{bmatrix} m_i & 0 & -m_i\left(\xi_{m_i}\sin\varphi_i + \eta_{m_i}\cos\varphi_i\right) \\ 0 & m_i & -m_i\left(-\xi_{m_i}\cos\varphi_i + \eta_{m_i}\sin\varphi_i\right) \\ -m_i\left(\xi_{m_i}\sin\varphi_i + \eta_{m_i}\cos\varphi_i\right) & -m_i\left(-\xi_{m_i}\cos\varphi_i + \eta_{m_i}\sin\varphi_i\right) & J_i + m_i\left(\xi_{m_i}^2 + \eta_{m_i}^2\right) \end{bmatrix} \tag{2.17}$$

and

$$A^i = \begin{Bmatrix} m_i(\xi_{m_i} \cos \varphi_i - \eta_{m_i} \sin \varphi_i)\dot{\varphi}_i^2 \\ m_i(\xi_{m_i} \sin \varphi_i + \eta_{m_i} \cos \varphi_i)\dot{\varphi}_i^2 \\ 0 \end{Bmatrix} \tag{2.18}$$

Using matrix notation, $M = \text{diag}(M^1, M^2, \ldots, M^n)_{3n \times 3n}$ and $A = [A^{1^T}, A^{2^T}, \ldots, A^{n^T}]^T$, Equation 2.15 becomes

$$M(q)\ddot{q} = A(q,\dot{q}) + Q(q,\dot{q},t) + \Phi_q^T(q,t)\lambda \tag{2.19}$$

Initial conditions for system motion are given as

$$\left.\begin{aligned} q(t_0) &= q^0 \\ \dot{q}(t_0) &= \dot{q}^0 \end{aligned}\right\} \tag{2.20}$$

where q^0 and $\dot{q}^0$ are consistent with constraints, i.e., q^0 satisfies Equation 2.8 and $\dot{q}^0$ satisfies

$$\Phi_q \dot{q} + \Phi_t = 0 \tag{2.21}$$

2.3 Integration of Constrained Equation of Motion by an Implicit Method

(a) Numerical Solution of Algebraic and Differential Equations

The method of numerically integrating differential equations presented in this and following chapters requires that they be reduced to first order form. This is accomplished by introducing the vector

$$w = \dot{q} \tag{2.22}$$

into Equation 2.19 to obtain

$$M(q)\dot{w} - A(q,w) - Q(q,w,t) - \Phi_q^T(q,t)\lambda = 0 \tag{2.23}$$

with initial conditions

$$\left.\begin{aligned} q(t_0) &= q^0 \\ w(t_0) &= w^0 = \dot{q}^0 \end{aligned}\right\} \tag{2.24}$$

where

$$\Phi_q(q^0,t_0)\, w\,(t_0) + \Phi_t(q^0,t_0) = 0 \tag{2.25}$$

In order to solve the differential equations of motion, numerical integration theory is used to obtain a set of approximations that is suitable for digital computation. Integration of the equations of motion is accomplished as a simultaneous solution of algebraic and differential equations. Standard approaches, however, are designed to solve systems of differential equations of the form

$$\dot{y} = f(y,t) \tag{2.26}$$

where y is an m-vector variable and f is an m-vector of functions. A modified approach is taken here that allows for the simultaneous solution of mixed algebraic and differential equations of the form

$$g(y,\dot{y},t) = 0 \tag{2.27}$$

where $\dot{y}$ may not appear in some of the equations. Before introducing the method to be used, it is instructive to review the standard approach applied to solving Equation 2.26.

The basic method of constructing approximate solutions is to place a grid of time points t_i, i = 1, ..., on the interval [0,T], where $h = t_{i+1} - t_i$ is the grid spacing. One then approximates the solution y(t) of Equation 2.26 at the time grid points as $y^i \approx y(t_i)$.

The basic approximating equation for stiff differential equations [35 and 36] is the Gear algorithm

$$y^{n+1} - h\beta_0 \dot{y}^{n+1} = \sum_{j=1}^{k} \alpha_j y^{n-j+1} \tag{2.28}$$

where the constants β_0 and α_j, j=1,...,k, called Gear's coefficients, are determined so that Equation 2.28 is exact for any polynomial solution of Equation 2.26 of degree up to k. Gear's coefficients [35 and 36] also have the property that the algorithm tends to be stable, even for stiff differential equations; i.e., differential equations with widely split eigenvalues.

A multistep formula that is used to solve Equation 2.27 is derived from Equation 2.28. One progresses from t_n to t_{n+1} by solving Equation 2.28, together with

$$g(y^{n+1}, \dot{y}^{n+1}, t_{n+1}) = 0 \tag{2.29}$$

Using the Newton formula to solve Equations 2.28 and 2.29, simultaneously, leads to

$$g_y^{(m)} \Delta y^{(m)} + g_{\dot{y}}^{(m)} \Delta \dot{y}^{(m)} = - g^{(m)} \tag{2.30}$$

where $\Delta y^{(m)} = y^{(m+1)} - y^{(m)}$ and m represents the iteration number. The time-step counter has been dropped here for notational simplicity. Substitution of Equation 2.30 into Equation 2.28, noting that the summation term of Equation 2.28 remains constant at each iteration, yields the corrector formulas

$$\left[\frac{1}{h\beta_0} g_{\dot{y}}^{(m)} + g_y^{(m)} \right] \Delta y^{(m)} = - g^{(m)} \tag{2.31}$$

$$\Delta \dot{y}^{(m)} = \frac{1}{h\beta_0} \Delta y^{(m)} \tag{2.32}$$

In the corrector formulas of Equation 2.31, the Jacobian matrix J is defined as

$$J = \left[\frac{1}{h\beta_0} g_{\dot{y}}^{(m)} + g_y^{(m)} \right] \tag{2.33}$$

If Equation 2.27 is of the form $P(y)\dot{y} + f(y,t) = 0$, which is the case here, then Equation 2.31 is of the form

$$\left[\frac{1}{h\beta_0} P^{(m)} + P_y^{(m)} \dot{y}^{(m)} + f_y^{(m)} \right] \Delta y^{(m)} = - g^{(m)} \tag{2.34}$$

The iterative corrector procedure is continued at each time step until all of the Newton differences $\Delta y^{(m)}$ are below a specified tolerance level. At each iteration, y and $\dot{y}$ are updated as

$$y^{(m+1)} = y^{(m)} + \Delta y^{(m)}$$

$$\dot{y}^{(m+1)} = \dot{y}^{(m)} + \Delta \dot{y}^{(m)} = \dot{y}^{(m)} + \frac{1}{h\beta_0} \Delta y^{(m)}$$

It is interesting to note that every element of $\dot{y}$ is obtained at each time step, even though many elements of y may not appear in any of the equations.

Since Gear's algorithm uses the Nordsieck vector representation, the predicted Nordsieck vector at the present time step is obtained by multiplying the Pascal triangle matrix times the Nordsieck vector at the previous time step [35]. Details of convergence criteria, error control, and procedures for this technique are discussed in references 20, 35, and 36.

(b) Sparse Matrix Algebra

The system of nonlinear algebraic and differential equations defined in the previous sections is loosely coupled. For two reasons, however, no attempt is made to obtain a smaller system of equations by reducing the number of equations through elimination of generalized coordinates. First, the Jacobian matrix formed by linearization of the coupled equations and used in iterative solution by Newton's method is sparse and can be very efficiently stored and decomposed. Secondly, the repetitive nature of the loosely coupled equations results in compact and efficient computer routines for evaluating the nonlinear equations and nonzero matrix entries. Recently developed sparse matrix algorithms [37] make both of these operations practical and desirable. It has been shown [37] that is is usually more efficient to solve large systems of sparse equations, rather than smaller systems with a greater percentage of nonzero entries.

Consideration of matrix sparsity is important to the speed of computation in problems of dynamic system analysis [35]. When less than 30% of the matrix entries are nonzero, it is inefficient to store the matrix as a two-dimensional array. Instead, the nonzero entries are stored in compacted form. The method most commonly used for compacting the data is to store the row and column indices of each nonzero-valued entry in the matrix in two vectors i and j, and its value in a third vector A. This is called "i-j" ordering and is a method of initially storing data from a user-supplied list of physical system elements.

Sparse matrix algorithms are most efficient when the nonzero matrix entries are stored in an organized manner. This usually implies that they are evaluated row-by-row or column-by-column. The previously mentioned repetitive matrix evaluation scheme is defeated by this requirement, since it usually results in the evaluation of small submatrices located at various positions throughout the matrix. To overcome this difficulty, a special permutation vector is generated from the row and column vectors describing the nonzero-valued positions. As each matrix entry is generated, it is directed to a specific location in the "A" vector by a permutation index. At completion of the matrix evaluation, all entries are stored exactly as if they had been evaluated in column order. A sparse matrix decomposition algorithm is then applied to the column-ordered matrix and the standard LU factorization [35] is accomplished. Full pivoting is not achieved, but the algorithm chooses, among the acceptable

pivot elements, the pivot that results in a minimum number of fills in the resulting L and U matrices. This is important for efficient execution of the forward and backward substitution phases, since an increased number of fills destroys the original matrix sparsity and results in excessive computer time.

CHAPTER 3

KINEMATIC ANALYSIS AND GENERALIZED COORDINATE PARTITIONING FOR DIMENSION REDUCTION IN ANALYSIS OF CONSTRAINED RIGID BODY SYSTEMS

3.1 Introduction

The implicit method of simultaneously solving systems of differential and algebraic equations of motion described in Chapter 2 is not desirable for several reasons. Implicit methods employ iterative techniques requiring a Jacobian matrix of the combined system of Equations 2.8, 2.22, and 2.23. For large systems, analytical expressions for derivatives of Equation 2.23 are required if program efficiency is to be achieved. Generalized forces, possibly discontinuous, may be represented by digitized data, from which it is difficult or impossible to provide derivatives, either analytically or numerically.

In the implicit method, the coordinates q, velocities w, and Lagrange multipliers λ of Equations 2.22 and 2.23 are treated as state variables, which are determined iteratively by the integration algorithm. The algorithm obtains q and w by integration, allowing it to maintain error control and accurate prediction of these variables. The algorithm uses the same predictor for Lagrange multipliers. Since generalized forces may be highly irregular or discontinuous,

irregular and discontinuous joint reaction forces are reflected in the Lagrange multipliers. The net result is a poor prediction of Lagrange multipliers, requiring more iterations to achieve convergence (or quite often divergence) in the corrector step. A reduction in time stepsize is then required to achieve better predicted values.

Stepsize reduction in the implicit algorithm is undesirable since it requires more computer time and often leads to failure of the algorithm to achieve a solution, even when extended precision computer arithmetic is used. The reason for corrector divergence is as follows: The time step h appears in the denominator of the expression multiplying the term $g_y^{(m)}$ in Equation 2.33, and as h gets smaller, these terms get larger and dominate the Jacobian matrix. The Jacobian matrix may thus become badly conditioned or singular. A badly conditioned matrix results in erroneous Newton differences associated with Lagrange multipliers, leading to divergence of the corrector.

The numerical method developed in the following chapters offers significant improvement over the implicit method of Chapter 2. Iterations are limited to the solution of constraint equations, thus requiring a much smaller Jacobian matrix. Since iteration only involves predicted variables with error control, convergence is much faster. In fact, the iteration step is frequently bypassed, because the constraint equations are satisfied by the predicted variables. Badly conditioned matrices, as described earlier, are avoided and single precision computer arithmetic can be used.

Another advantage of the proposed method is that a minimal set of differential equations is identified by the program and solved by an explicit integration method. This avoids a Jacobian matrix involving derivatives of generalized forces.

3.2 System Constraints

(a) Classification of System Coordinates

The ultimate goal of this research is to develop analysis techniques that use system constraints to improve numerical efficiency and accuracy, while at the same time reducing the preprocessing requirements of the user. To this end, an understanding of what constitutes a constraint is essential. According to Webster [38] a constraint is confinement; restriction; force; compulsion; or coercion. In the tradition of mechanics, a constraint is any condition that reduces the freedom of a system. Another notion is "something that limits motion".

In the development of Chapter 2, a system was assumed to be composed of n rigid bodies in a plane, each with three degrees of freedom. Algebraic constraint equations were also formulated and reaction forces at the constraint surfaces were introduced into the system differential equations of motion, through a vector of Lagrange multipliers. The equations of motion and constraint equations were solved iteratively and no account was taken of the number of system degrees of freedom. Mathematically, this system has 3n degrees of

freedom and the algebraic constraints/constraint reaction force system represents a restriction on admissible movements.

The formulation of Chapter 2 treats every coordinate as independent, hence the system has 3n generalized coordinates. A formulation reduces the number of degrees of freedom when the algebraic constraints/constraint reaction force balance is used to eliminate generalized coordinates from the system equations of motion. Application of constraint relations to mathematically reduce the number of degrees of freedom introduces a new problem of identifying independent and dependent variables. If constraint equations mathematically eliminate r degrees of freedom, there are (3n-r) independent generalized coordinates and the remaining r coordinates become dependent position coordinates. For notational purposes, let the symbol u designate dependent variables and let v designate independent generalized coordinates. With this notation, $q = [u^T, v^T]^T$.

Partitioning of q into u and v must be done in a manner that does not degrade program efficiency or increase numerical errors. If the partitioning were done on a random basis, there are $3n!/r!(3n-r)!$ possible combinations, most of which would be mathematically unacceptable. Wells [39] stated, when discussing the selection of independent coordinates for simple problems: "It is a well known fact that certain coordinates may be more suitable than others. Hence the quantities chosen in any particular case are those which

appear to be the most advantageous for the problem in hand. The final choice depends largely on insight and experience." This approach is also taken in another well developed and documented planar rigid body modelling program [13]. However, in this case, steps are taken to circumvent some of the numerical difficulties associated with bad choices of independent variables. When systems become large and complex, geometrical insight and experience quite often fail and mathematical techniques must be employed.

To the author's knowledge, the first organized method for selection of a good, and possibly best, set of independent generalized coordinates for large scale systems was developed by Sheth [40, 16]. Although he does not prove that the selected set is best, he does provide numerous arguments, based on geometrical considerations and example test problems, that indicate the set has desirable features of a properly driven kinematic mechanism. This approach has sound mathematical appeal and will be developed as the tool for efficient dimension reduction of constrained systems.

(b) Iterative Solution of System Constraint Equations

The system constraint equations developed in Chapter 2 are written symbolically in the form

$$\Phi(q,t) = 0 \tag{3.1}$$

where all equations may not be independent, but they are consistent. That is, the Jacobian matrix Φ_q with m rows and $3n$ columns has row rank $0 \leq r \leq m$ and $r \leq 3n$. There are one or more nonsingular submatrices of Φ_q of rank r. Gauss-Jordan reduction of the matrix Φ_q, with double pivoting, defines a partitioning of $q = [u^T, v^T]^T$ such that Φ_u is a submatrix of Φ_q of rank r, whose columns correspond to elements u of q, and Φ_v is a submatrix of Φ_q whose columns correspond to elements v of q. Furthermore, the matrix Φ_u has ideal numerical properties associated with double pivoting.

The subprogram described in Appendix A determines the rank of Φ_q and its linearly independent rows and columns. It performs Gaussian elimination with row and column permutations that bring independent rows and columns to the top and left of the matrix, until the remaining submatrix in the lower-right corner is null (all of its elements are less than some specified tolerance). In addition, the matrix is decomposed into factors of the form

$$\begin{bmatrix} L & 0 \\ LR & 0 \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} U & UR \\ 0 & 0 \end{bmatrix}$$

which are superimposed onto the original matrix. The factor products

$$\begin{bmatrix} L \cdot U & L \cdot UR \\ LR \cdot U & LR \cdot UR \end{bmatrix}$$

represent, in permuted form, the original matrix Φ_q; specifically, $L \cdot U = \Phi_u$ and $L \cdot UR = \Phi_v$. The columns of U and UR determine a

permutation of q into dependent (u) and independent (v) variables, respectively. The rows of L and LR determine a permutation of constraint equations $\Phi = 0$ into independent Φ^1 and dependent Φ^2 equations, respectively. In addition, the respective lower and upper triangular matrices L and U are nonsingular.

In the following chapters it is often convenient, for clarity, to express matrix equations in the above permuted and factored forms, rather than in terms of the original matrix Φ_q.

Some additional comments can be made concerning the matrix Φ_q and its corresponding submatrices. First of all, Φ_q may be null. That is, there may be no constraints in the system, in which case $v = q$ and u is null. This special case requires no further analysis and need not be considered here.

If Φ_v is null there are no independent generalized coordinates and $u = q$. In this case, in view of Equation 3.1, the system is either a structure $\Phi = \Phi(q)$ or its geometry is completely determined as a function of time, $\Phi = \Phi(q,t)$.

If roots of Equation 3.1 are to be found, an iterative procedure such as Newton's method is required. With an initial estimate $q(0)$ of the system configuration at time t_0, the matrix $\Phi_q(q(0),t_0)$ is evaluated. The iterative equation is then

$$\Phi_q[q(j),t_0]\,[q(j+1) - q(j)] = -\Phi[q(j),t_0],\ j = 0,1,\ldots \quad (3.2)$$

where j is an iteration counter. The solution of Equation 3.1 requires the determination of q(j+1) in Equation 3.2. If Equation 3.1 is consistent, there is at least one solution. If the rank is less than 3n there are an infinite number of solutions.

Assume that the matrix has been decomposed by the subprogram described in Appendix A. Equation 3.2 may then be written in permuted form, in terms of the previously mentioned matrix factors and partitioning, as

$$L \cdot U \cdot [u(j+1) - u(j)] + L \cdot UR \cdot [v(j+1) - v(j)]$$

$$= - \Phi^1[q(j),t_0] \tag{3.3}$$

$$LR \cdot U \cdot [u(j+1) - u(j)] + LR \cdot UR \cdot [v(j+1) - v(j)]$$

$$= - \Phi^2[q(j),t_0] \tag{3.4}$$

Noting that v are independent generalized coordinates, one may set $v(j+1) = v(j)$ and Equation 3.3 reduces to

$$L \cdot U \cdot [u(j+1) - u(j)] = - \Phi^1 \tag{3.5}$$

Since matrices L and U are nonsingular and triangular, simple forward elimination and back substitution determines $u(j+1)$. Equation 3.4 is not required in the intermediate steps of

determining roots of Equation 3.1, but may be used to test for consistency of constraints at the final step. That is, $||\Phi^2||$ must be less than a specified tolerance, if redundant constraints are to be satisfied.

Kinematic analysis of constrained systems requires that bodies move consistent with constraints. If the system is moved from position q^1 at t_1 to a nearby position q^2 at t_2, consistent with constraints, then $\Phi(q^1,t_1) = \Phi(q^2,t_2) = 0$. If $\Phi(q^2,t_2)$ is expanded in a Taylor series about q^1 and t_1 one has

$$\Phi(q^2,t_2) = \Phi(q^1,t_1) + \Phi_q \Delta q + \Phi_t \Delta t + \frac{1}{2} (\Phi_q \Delta q)_q \Delta q$$

$$+ \ldots = 0 \tag{3.6}$$

where $\Delta q = q^2 - q^1$ and $\Delta t = t_2 - t_1$. If Δq and Δt are small, second and higher order terms of Equation 3.6 can be neglected, yielding a linear predictor of q^2, of the form

$$\Phi_q (q^2 - q^1) = - \Phi_t \Delta t \tag{3.7}$$

Comparing Equations 3.7 and 3.2 and noting the permuted form of Equations 3.3 and 3.4, one can express Equation 3.7 as

$$L \cdot U \cdot (u^2 - u^1) + L \cdot UR \cdot (v^2 - v^1) = - \Phi_t^1 \Delta t \tag{3.8}$$

$$LR \cdot U \cdot (u^2 - u^1) + LR \cdot UR \cdot (v^2 - v^1) = - \Phi_t^2 \Delta t \qquad (3.9)$$

Since L and U are nonsingular and with v^2 and t_2 specified, Equation 3.8 determines u^2. It is most efficient to solve Equation 3.8 in two steps, since the matrix product L · UR is not calculated by the subprogram. First, solve

$$L \cdot U \cdot u^* = - \Phi_t^1 \Delta t \qquad (3.10)$$

by forward elimination and back substitution and then use the precalculated matrix H to obtain

$$u^2 - u^1 = H \cdot (v^2 - v^1) + u^* \qquad (3.11)$$

Equation 3.3 is then used to iteratively correct u^2, to satisfy Equation 3.1.

The matrix product L · UR of Equation 3.8 is never evaluated, since the matrix

$$H = - U^{-1} \cdot L^{-1} \cdot L \cdot UR = - U^{-1} \cdot UR$$

is required in Equation 3.1. In fact all subsequent equations involving the matrix L · UR ultimately require the matrix H for solution, so for convenience, H is calculated and stored in the space originally occupied by UR.

To avoid excessive corrector iterations, it is desirable that Equation 3.1 predict u^2 with small error. Since Δq was assumed small and Δv is fixed, Δu should be small. Gaussian elimination, using full row and column pivoting to partition q into u and v, accomplishes this. Equation 3.1 may be written simply as

$$\Delta u = H\ \Delta v + u^*$$

As noted in Appendix A, the largest matrix element is permuted to the diagonal position at each intermediate step, when performing Gaussian elimination with full row and column pivoting. This process tends to maximize the magnitude of diagonal elements of U and to minimize the magnitude of elements of UR. Noting that $H = - U^{-1} \cdot UR$, one would expect that the magnitude of elements in H will be substantially less than 1 and that the magnitude of elements in Δu will be substantially less than the magnitude of elements in Δv.

This argument could be strengthened if a reliable estimate for the condition number of U, cond(U), were available. In general, Gaussian elimination with full row and column pivoting yields well conditioned matrices [41]. However, one can find examples in which it does not. If cond(U) and the norm of U are known, the norm of U^{-1} is

$$||U^{-1}|| = \text{cond}(U)/||U|| \qquad (3.12)$$

The norm of U is, by construction, large since the largest elements of Φ_q were permuted into it. If cond(U) is small (on the order of 1), then by Equation 3.12, $||U^{-1}|| \approx 1/||U||$.

Since the elements of U dominate those in UR, it is reasonable to expect that $||U|| \gg ||UR||$ and $||U^{-1}|| \cdot ||UR|| \ll 1$, by the assumption of a well conditioned U matrix. Finally, $||H|| = ||- U^{-1} \cdot UR|| < ||U^{-1}|| \cdot ||UR|| \ll 1$, supports the contention that $||\Delta q||$ is reduced by proper selection of u and v.

3.3 Kinematic Velocity and Acceleration Analysis

(a) Velocity Analysis

For kinematic analysis, if one specifies independent generalized coordinates as functions of time, independent velocities and accelerations can also be expressed as functions of time.

The remaining dependent velocities and accelerations can then be determined by differentiating the constraint equations with respect to time. Differentiating Equation 3.1 with respect to time yields

$$\Phi_q \dot{q} + \Phi_t = 0 \tag{3.13}$$

Comparing Equations 3.13 and 3.2, equations similar to Equations 3.3 and 3.4 can be expressed as

$$L \cdot U \cdot \dot{u} + L \cdot UR \cdot \dot{v} = - \Phi_t^1 \tag{3.14}$$

$$LR \cdot U \cdot \dot{u} + LR \cdot UR \cdot \dot{v} = - \Phi_t^2 \tag{3.15}$$

The solution of Equation 3.14 determines $\dot{u}$ and Equation 3.15 is automatically satisfied by the consistency requirement of Equation 3.4.

Similar to the method of solving Equation 3.8, first solve

$$L \cdot U \cdot \dot{u}^* = - \Phi_t^1 \tag{3.16}$$

by forward elimination and back substitution and then use the precalculated matrix H to obtain

$$\dot{u} = H \dot{v} + \dot{u}^* \tag{3.17}$$

Inspection of Equation 3.17 reveals that the matrix H relates dependent velocities to independent velocities and this matrix is generally referred to as an influence coefficient matrix. From a geometrical point of view, a properly driven mechanical system is one in which the ratios of output to input velocities are minimized [40, 16]. In addition, the system is driven with the greatest mechanical advantage.

Recall that the method of decomposing Φ_q determined H, such that Δu was reduced in Equation 3.11, hence $\dot{u}$ is also reduced in Equation 3.17. A further advantage of minimizing velocity ratios is that errors in independent velocities are not amplified when calculating dependent velocities. This will be important when solving the system differential equations of motion in Chapter 4.

(b) Acceleration Analysis

Having determined all system position and velocity variables, one may proceed to determine dependent accelerations. Equation 3.13 may be differentiated with respect to time to obtain

$$\Phi_q \ddot{q} + (\Phi_q \dot{q})_q \dot{q} + 2\Phi_{qt}\dot{q} + \Phi_{tt} = 0 \tag{3.18}$$

For convenience, denote the three known terms in Equation 3.18 as

$$C(q,\dot{q},t) = (\Phi_q \dot{q})_q \dot{q} + 2\Phi_{qt}\dot{q} + \Phi_{tt} \tag{3.19}$$

and partition Equation 3.18 as

$$L \cdot U \cdot \ddot{u} + L \cdot UR \cdot \ddot{v} = - C^1 \tag{3.20}$$

$$LR \cdot U \cdot \ddot{u} + LR \cdot UR \cdot \ddot{v} = - C^2 \tag{3.21}$$

The independent generalized accelerations $\ddot{v}$ are presumed known, so Equation 3.20 determines $\ddot{u}$ and Equation 3.21 is automatically satisfied by the consistency requirement of Equation 3.4.

Similar to Equation 3.8, Equation 3.20 is solved in two steps

$$L \cdot U \cdot \ddot{u}^* = - C^1 \tag{3.22}$$

by forward elimination and back substitution. Also,

$$\ddot{u} = H\,\ddot{v} + \ddot{u}^{*} \tag{3.23}$$

where the precalculated influence coefficient matrix H is used. Note that errors in independent generalized accelerations $\ddot{v}$ are reduced, in calculating the dependent accelerations $\ddot{u}$, because of the small norm of H.

3.4 Kinetostatics of Constrained Systems

The analysis method of Section 3.2 allows one to identify an ideal set of independent generalized coordinates, from which to drive an arbitrary, constrained multiple-degree-of-freedom system. Kinetostatic analysis may then be applied, using the method of Section 3.3, to determine all forces in the system, including inertial forces, generalized joint reaction forces, and the generalized driving forces required to achieve the specified system dynamics. One could then envision the same system reacting to this set of generalized driving forces. In this case, a system of differential equations of motion, solved by the methods of Chapter 4 will determine the same dynamic response. The first system is displacement driven, requiring specified generalized coordinates, and the second system is force driven, requiring specified generalized forces. The two examples demonstrate a duality between specified generalized coordinates and specified generalized forces.

Using the permutations of the previous sections, equations of motion for constrained systems derived in Chapter 2 may be partitioned into

$$U^T \cdot L^T \cdot \lambda^1 + U^T \cdot LR^T \cdot \lambda^2 = M^{uu}\dot{u} + M^{uv}\dot{v}$$

$$- (A^u + Q^u) \tag{3.24}$$

$$UR^T \cdot L^T \cdot \lambda^1 + UR^T \cdot LR^T \cdot \lambda^2 = M^{vu}\dot{u} + M^{vv}\dot{v}$$

$$- (A^v + Q^v) \tag{3.25}$$

Lagrange multipliers λ^1 and λ^2 are associated with independent constraint equations $\Phi^1 = 0$ and redundant or dependent constraint equations Φ^2, respectively. The multipliers λ^2 are not determined by Equations 3.24 and 3.25, but must be obtained from other methods, such as local body deformation models that specify relations between λ^1 and λ^2.

Let

$$F^u = U^T \cdot L^T \cdot \lambda^1 + U^T \cdot LR^T \cdot \lambda^2 \tag{3.26}$$

and

$$F^v = UR^T \cdot L^T \cdot \lambda^1 + UR^T \cdot LR^T \cdot \lambda^2 \tag{3.27}$$

Then, by combining Equations 3.26 and 3.27 with $H = - U^{-1} \cdot UR$, it is easy to show that

$$F^v = - H^T F^u \tag{3.28}$$

holds for any value of λ^1 and λ^2. Assuming that redundant constraints are consistent, they can be eliminated without effecting the system dynamics, which is then equivalent to setting λ^2 to zero in Equations 3.24 and 3.25.

A displacement driven system with specified generalized coordinates has corresponding unknown generalized forces. In this case, let the generalized force vector Q^v be composed of known Q^{v*} and unknown R^v forces

$$Q^v = Q^{v*} + R^v \tag{3.29}$$

Generalized constraint reaction forces become

$$F^u = U^T \cdot L^T \cdot \lambda^1 = M^{uu}\dot{u} + M^{uv}\dot{v} - (A^u + Q^u) \tag{3.30}$$

and the unknown generalized driving forces become

$$R^v = M^{vu}\dot{u} + M^{vv}\dot{v} - (A^v + Q^{v*}) + H^T F^u \tag{3.31}$$

It is not necessary to evaluate λ^1 to determine the generalized constraint reaction forces.

Equations 3.30 and 3.31 may be made more tractable by substituting Equation 3.23 to eliminate dependent accelerations, thus

$$F^u = (M^{uu}H + M^{uv})\dot{v} - (A^u + Q^u - M^{uu}\dot{u}^*) \tag{3.32}$$

$$R^v = (M^{vu}H + M^{vv})\dot{v} - (A^v + Q^{v^*} - M^{vu}\dot{u}^*) + H^T F^u \tag{3.33}$$

If forces F^u are not desired, one may combine Equations 3.32 and 3.33 to eliminate F^u, obtaining

$$R^v = [M^{vv} + M^{vu}H + H^T(M^{uv} + M^{uu}H)]\dot{v} - (A^v + Q^{v^*} - M^{vu}\dot{u}^*)$$

$$- H^T(A^u + Q^u - M^{uu}\dot{u}^*) \tag{3.34}$$

Equation 3.34 thus relates generalized driving forces R^v to specified generalized coordinates v, where velocities and accelerations are assumed known.

The kinetostatics problem with full consideration of all forces in the system may now be solved. The basic steps are, presuming equations and initial coordinate estimates have been given:

Step 1. Identify dependent and independent variables. Decompose the matrix Φ_q, using the method of Appendix A, to determine a partitioning of q into u and v.

Step 2. Solve the constraint equations. Specify v and iteratively solve Equation 3.1, using Equation 3.5. Check consistency of equations $\Phi^2 = 0$.

Step 3. Calculate system velocities. Specify $\dot{v}$ and determine $\dot{u}$ from Equations 3.16 and 3.17.

Step 4. Calculate system accelerations. Specify $\ddot{v}$, evaluate C^1, and determine $\ddot{u}$ from Equations 3.22 and 3.23. Dependent accelerations $\ddot{u}$ are not required if Equations 3.32, 3.33, or 3.34 are used in Step 5.

Step 5. Determine generalized constraint reaction forces F^u and system driving forces R^v. Evaluate A and Q and determine F^u from Equation 3.32 and R^v from Equation 3.33 or Equation 3.34. Individual constraint reaction forces are obtained by first solving

$$U^T L^T \lambda^1 = F^u$$

(obtained from Equation 3.26 with $\lambda^2 = 0$) for λ^1. Note that

$$F^u = \Phi_u^T \lambda^1$$

and

$$F^v = \Phi_v^T \lambda^1$$

Each column entry in a given row of Φ_u^T (Φ_v^T) times the corresponding row entry of λ^1 determines the generalized constraint reaction force contribution to the generalized coordinate associated with the given row of Φ_u^T (Φ_v^T). Hence, one may calculate reaction forces at any constraint in the system.

3.5 Sparse Matrix Considerations

As noted in Chapter 2 the nodal method of constraint formulation yields loosely coupled equations with a corresponding sparse Jacobian matrix. If the system has a large number of constraints, sparse matrix manipulation algorithms will be efficient. While the equations derived in previous sections still apply, they may be expressed in different forms to avoid unnecessary calculation of sparse matrix products.

Sparse matrix algorithms that factor matrices are not suitable for determining the partitioning of coordinates q into u and v. They employ a partial pivoting strategy and to maintain efficiency, usually do not select the largest pivotal elements. The matrix Φ_u so identified will have a smaller norm than the corresponding matrix identified by full pivoting. The corresponding matrix $H = -\Phi_u^{-1}\Phi_v$ will then have a larger norm. Therefore, Gaussian elimination with full pivoting is periodically applied to the full matrix representation of Φ_q solely for the purpose of determining the ordering of q into u and v. Once established, a matrix $\bar{I}$ is appended to Φ_q to form a modified matrix $\bar{\Phi}_q$ of rank 3n such that when $\bar{\Phi}_q$ is permuted, corresponding to

the permutation of Φ_q into Φ_u and Φ_v it is of the form

$$\left[\begin{array}{c|c} \Phi_u & \Phi_v \\ \hline 0 & I \end{array}\right]$$

where I is an identity matrix. The matrix $\bar{I}$ is readily constructed from the permutation vectors generated by the algorithm described in Appendix A.

An algorithm employing sparse matrix techniques for a displacement driven system with full consideration of inertial and externally applied loads is given in the following steps:

Step 1. Periodically identify dependent and independent variables by decomposing the matrix Φ_q, using the method of Appendix A. Construct the matrix $\bar{\Phi}_q$ by appending the matrix $\bar{I}$ to Φ_q.

Step 2. If a first order prediction of dependent variables (u^1) to an adjacent position (u^2) is desired, append to Equation 3.7,

$$\Phi_q \Delta q = - \Phi_t \Delta t$$

the equation

$$\Delta v = v^2 - v^1$$

which in matrix form becomes

$$\bar{\Phi}_q \Delta q = \begin{bmatrix} \Phi_t \Delta t \\ v^2 - v^1 \end{bmatrix} \qquad (3.35)$$

The matrix $\bar{\Phi}_q$ of rank r = 3n is then factored by sparse matrix algorithms and Δq is determined by Equation 3.35.

Step 3. The predicted dependent variables u are now corrected iteratively until the constraint equations are satisfied by appending to Equation 3.2,

$$\Phi_q \Delta q = - \Phi$$

the requirement that independent variables remain fixed

$$\Delta v = 0$$

which in matrix form becomes

$$\bar{\Phi}_q \Delta q = \begin{bmatrix} - \Phi \\ 0 \end{bmatrix} \tag{3.36}$$

Dependent variables are iteratively corrected until the independent constrinats are satisfied and any dependent constraint equations are then checked for consistency.

Step 4. Analysis of displacement driven systems requires that independent velocities be specified. Denoting these as $\dot{v}^s$, write Equation 3.13 as

$$\Phi_q \dot{q} = - \Phi_t$$

and append the equations

$$\dot{v} = \dot{v}^s$$

which in matrix form becomes

$$\bar{\Phi}_q \dot{q} = \begin{bmatrix} -\Phi_t \\ \dot{v}^s \end{bmatrix} \tag{3.37}$$

Equation 3.37 then determines $\dot{q}$.

Step 5. Combine Equation 3.18 and $\ddot{v} = \ddot{v}^s$ to form the matrix equation

$$\bar{\Phi}_q \ddot{q} = \begin{bmatrix} -C \\ v^s \end{bmatrix} \tag{3.38}$$

where $C(q, \dot{q}, t)$ is given by Equation 3.19. Solve for accelerations $\ddot{q}$.

Step 6. Using Equation 2.23 and noting from Equation 3.29 that unknown generalized forces R^v associated with independent generalized coordinates must be determined, one has

$$\Phi_q^T \lambda + \bar{I}^T R_v = M\ddot{q} - (A + Q)$$

which in matrix form becomes

$$\bar{\Phi}_q^T \begin{bmatrix} \lambda \\ R^v \end{bmatrix} = M\ddot{q} - (A + Q) \tag{3.39}$$

As noted in Section 3.4, Lagrange multipliers corresponding to redundant constraints, are not determined by Equation 3.39. However, if redundant constraints are consistent, their corresponding multipliers can be set to zero. The remaining multipliers and unknown forces R^v are then determined by Equation 3.39 and all forces in the system are determined.

Equations 3.35 to 3.39 employ the same sparse matrix $\bar{\Phi}_q$ or $\bar{\Phi}_q^T$, thus only one matrix factorization is required for each execution of steps 2 to 6. Matrix and vector products are also indicated for evaluation of the right hand side of Equations 3.38 and 3.39. However, these matrices are very sparse and analytical expressions are written directly for the indicated products requiring no additional matrix manipulation.

To maintain program efficiency the full matrix decomposition of Step 1 should not be performed too frequently, yet often enough to avoid numerical problems associated with nearly singular Φ_u matrices. Available methods for monitoring the condition number of Φ_u require more time than is required for its original determination. Monitoring velocity ratios, however, may be an inexpensive, though not entirely reliable, technique for determining the need for a new Φ_u matrix. Equation 3.13 may be partitioned and written as

$$\dot{u} = - \Phi_u^{-1} [\Phi_v \dot{v} + \Phi_t]$$

To approximate the amplifying ability of Φ_u^{-1} one could evaluate the number

$$K_i = \sqrt{\Sigma\, \dot{u}^2/(\Sigma\, \dot{v}^2 + \Sigma\, \Phi_t^2)}$$

following the initial determination of Φ_u, $(i = 0)$ and at each successive application of the sparse matrix code $(i > 0)$. When the estimate of increase in matrix norm K_i/K_0 exceeds a specified level, a need for a new partitioning of Φ_q is then indicated.

CHAPTER 4

CONSTRAINED EQUILIBRIUM AND DYNAMIC ANALYSIS

4.1 Introduction

In Chapter 3, differential equations of motion are developed for a general constrained planar rigid body system. It was demonstrated that the complete system state is determined when all independent generalized coordinates, velocities, and acclerations are specific functions of time. These equations, however, must be numerically integrated when independent variables are allowed to move under the influence of applied force. Therefore, this chapter is devoted to development of an efficient numerical integration algorithm utilizing kinematic properties of constraints to obtain a minimal system of differential equations of motion.

Efficiency is achieved by minimizing the number of differential equations to be solved, which improves numerical integration performance. Paul [42] has indicated that the minimum number of first order differential equations formulated by any of the existing general purpose programs is 2 dof (degree of freedom). Only 2 dof first order equations are required by the method described in this chapter.

4.2 Equilibrium Analysis

Transient, dynamic analysis of complex constrained mechanical systems is often initiated from a position of partial or complete equilibrium. Static equilibrium analysis is often required prior to initiation of transient analysis of vehicular systems, because the nonlinear characteristics of suspension elements make it difficult to estimate equilibrium configurations. Partial equilibrium analysis is required when one or more of the bodies in a system have no equilibrium position or when internal forcing elements such as springs, are to be given initial lengths differing from their equilibrium lengths. In this case, temporary constraints are placed on or between bodies prior to equilibrium analysis and released prior to transient analysis.

Various methods may be employed to solve equilibrium equations. Since they are highly nonlinear, iterative techniques are required. One iterative technique that is often employed is Newton's method for finding roots of nonlinear algebraic equations. Newton's method may converge to unstable equilibrium configurations, or diverge, if poor initial estimates of configuration are given. However, it is easy to implement and has been used to find equilibrium of systems with many degrees of freedom.

The implicit numerical integration method for transient analysis of Chapter 2 requires the Jacobian matrix of the equations of motion for solution of Equation 2.23. Noting that velocities and

accelerations are zero for static equilibrium, Equation 2.23 expresses equilibrium equations as

$$\left.\begin{aligned} \Phi_q^T(q, t_o)\,\lambda + Q(q, t_o) &= 0 \\ \Phi(q, t_o) &= 0 \end{aligned}\right\} \tag{4.1}$$

and the corresponding matrix iterative equation is

$$\begin{bmatrix} (\Phi_q^T\lambda + Q)_q & \Phi_q^T \\ \Phi_q & 0 \end{bmatrix} \begin{bmatrix} \Delta q(j) \\ \Delta\lambda(j) \end{bmatrix} = - \begin{bmatrix} \Phi_q^T\lambda + Q \\ \Phi \end{bmatrix} \tag{4.2}$$

This method, requiring the simultaneous determination of q and λ, is undesirable because it involves the repetitive solution of a large system of equations and a reasonable estimate of the vector λ must be given, since it appears in the matrix. Poor estimates of λ may lead to a badly conditioned or singular matrix in Equation 4.2 and divergence of Newton's method. For this reason, iterative techniques that do not involve Lagrange multipliers are desirable.

In Chapter 3 a method for identifying the independent generalized coordinate vector v was developed and a reduced system of differential equations of motion, Equation 3.34, was derived. Noting

that velocities and accelerations are zero for static equilibrium, Equation 3.34 yields

$$R^v = - H(q, t_o)^T Q(q, t_o)^u - Q(q, t_o)^{v*} \tag{4.3}$$

where

$$\Phi(q, t_o) = 0 \tag{4.4}$$

The forces R^v must be applied to the independent or specified coordinates v^s in a displacement driven system. In an equilibrium situation the variables v are free and the forces R^v must be zero. A condition for static equilibrium is then

$$R^v = 0 \tag{4.5}$$

Equations 4.4 and 4.5 may be solved iteratively, by noting that $R^v = R^v(u, v)$. Therefore, with du = Hdv, the iterative equation

$$\frac{dR^v}{dv} \Delta v = [R^v_v + R^v_u H]\Delta v = - R^v \tag{4.6}$$

determines the solution of Equation 4.5. The matrix in Equation 4.6 is evaluated by numerical differencing, because 1) the matrix H and

generalized forces Q appear in Equation 4.3, 2) analytical expressions for their partial derivatives are difficult to obtain, and 3) the derivatives are not required for subsequent transient analysis.

4.3 Initial Conditions

Obtaining suitable initial conditions for transient analysis poses problems when constrained mechanical systems have many degrees of freedom. Often one desires to specify initial conditions on various coordinates and their first derivatives or initial conditions on first derivatives of other coordinates. The independent variable set identified by factoring the constraint Jacobian matrix may not agree with the above selected variables so some initial conditions would be changed when satisfying constraints. The method described in Section 3.6 allows one to hold selected coordinates and velocities at their specified values by appending temporary constraints. The only requirement is that these constraints be consistent with other constraints. This method is employed below. The following seven steps outline an algorithm for achieving suitable initial conditions, from a starting configuration that is consistent with constraints or from a partial or full equilibrium configuration:

Step 1. Specify an initial estimate q^0 of the constrained system configuration. This estimate must be near the expected final configuration to insure convergence. If static equilibrium is to be obtained, the estimate must be near the desired equilibrium

configuration to avoid convergence to alternate stable or unstable configurations, or divergence. Set $q = q^0$.

Step 2. If some elements of q are to remain fixed during initial assembly of the constrained system or during static equilibrium, for example in partial equilibrium, append to Equation 4.4 additional algebraic constraints of the form $q_j - q_j^0 = 0$, where j identifies the variables to be held fixed. Other types of algebraic relations between coordinates can also be formulated as needed. These constraints will be removed prior to step 6.

Step 3. Evaluate the modified constraint matrix Φ_q (including constraints of step 2) and factor, as described in Appendix A, to determine a submatrix Φ_u of maximal rank and to determine a partitioning of q into u and v (dependent and independent coordinates, respectively). If Φ_q does not have full row rank, then too many constraints or inconsistent constraints are specified. Therefore some equations are dependent and they can be temporarily ignored in the iterative procedure of Step 4.

Step 4. Keeping the independent variables v fixed, iterate with Equation 3.5 to solve the independent constraint equations of step 2, until $||\Phi|| < \epsilon$, where ϵ is a specified closure tolerance level. If there are dependent constraint equations, check them for violation. When dependent constraints are violated, inconsistent constraint conditions have been specified and a meaningless solution may result. If static equilibrium is not desired, go to step 6.

Step 5. Determine a new estimate for v from Equation 4.6 that reduces $||R^v||$. If $||R^v|| > \delta$, a specified force unbalance tolerance, return to step 3. If the iteration in steps 3 to 5 diverges, either inconsistent constraints have been imposed or initial estimates of system configuration are in error.

Step 6. Specify initial estimates for velocities $\dot{q}^0$ and set $\dot{q} = \dot{q}^0$. The velocities $\dot{q}$ are adjusted to satisfy Equation 3.13, evaluated at q^0 from the above steps. If some elements of $\dot{q}$ are to remain fixed at $\dot{q}^0$, append to Equation 3.13 additional equations of the form $\dot{q}_k - \dot{q}_k^0 = 0$, where k identifies the variables to be held fixed. Other types of relations between velocities can also be formulated as needed. These constraints will be removed prior to step 1 of transient analysis.

Step 7. Factor the modified matrix of Equation 3.13 as described in Appendix A, to determine a partitioning of $\dot{q}$ into $\dot{u}$ and $\dot{v}$ (not necessarily the same partitioning as in step 3). Calculate $\dot{u}$ using the above partitioning in Equations 3.16 and 3.17. If the above matrix lacks full row rank, some equations are dependent and must be checked for consistency. If dependent equations are not satisfied, inconsistent constraints on velocity are specified and a meaningless solution may result. Otherwise, a consistent set of initial conditions is now available for transient analysis.

4.4 Numerical Integration of the Equations of Motion

The reduced system of differential equations of motion, derived in Chapter 3, were expressed in the form

$$R^v = [M^{vv} + M^{vu} H + H^T(M^{uv} + M^{uu} H)]\ddot{v} - (A^v + Q^{v*} - M^{vu} \ddot{u}^*)$$

$$- H^T(A^u + Q^u - M^{uu} \ddot{u}^*) \tag{4.7}$$

Noting that the independent variables v are free, the driving forces R^v are zero in Equation 4.7. Therefore Equation 4.7 can be written as

$$[M^{vv} + H^T M^{uv} + (M^{vu} + H^T M^{uu})H]\dot{v} = A^v + Q^v + H^T(A^u + Q^u)$$

$$- (M^{vu} + H^T M^{uu})\ddot{u}^* \tag{4.8}$$

In order to integrate Equation 4.8, it is helpful to write it in first order form. To do this, define the vector s of independent velocities as

$$\dot{v} = s \tag{4.9}$$

and write the total vector $\dot{q}$ of generalized velocities in terms of s, using Equations 3.16 and 3.17, as

$$w(s) = \dot{q}(s) = \begin{bmatrix} \dot{u} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} Hs + \dot{u}^* \\ s \end{bmatrix} \tag{4.10}$$

where $L \cdot U \cdot \dot{u}^* = - \Phi_t^1$. With this notation Equation 3.19 becomes a function of w,

$$C = (\Phi_q w)_q w + 2\,\Phi_{qt}\, w + \Phi_{tt}$$

Equation 3.22 is used to evaluate $\ddot{u}^*$, and Equation 4.8 becomes

$$[M^{vv} + H^T M^{uv} + (M^{vu} + H^T M^{uu})H]\dot{s} = A^v + Q^v + H^T(A^u + Q^u)$$

$$- (M^{vu} + H^T M^{uu})\ddot{u}^* \qquad (4.11)$$

Decomposing the above mass matrix as described in Appendix A, one may solve Equation 4.11 for $\dot{s}$. Equations 4.9 and 4.11 now form a system of $6n - 2r = 2$ dof first order ordinary differential equations where n is the number of bodies in the system and r is the number of dependent variables in u.

The selection of numerical integration algorithms for dynamic analysis should be based on the type of differential equations to be solved. Since most mechanical systems are non-stiff, an explicit/implicit-predictor/corrector method may be employed [28]. Predictor/corrector algorithms require fewer function evaluations than Runga-Kutta methods and allow interpolation between solutions with no additional function evaluations. This is important, since the computational overhead in evaluating the reduced system of differential equations is significant. The explicit/implicit-predictor/corrector algorithm DE/STEP, INTRP [28] is used, since it is well developed and easy to adapt to the equation requirements. In addition, the

integration algorithm DE estimates relative stiffness and terminates execution if the system becomes too stiff.

A numerical integration algorithm based on the development in previous sections and the explicit integration method is now presented. The following steps outline the numerical integration algorithm.

Let i, initially 0, be an indicator for the current time step; i.e., i = 0 implies $t = t_0$. Steps 1 to 5 are satisfied when starting, from results of the preceding algorithm, so start with step 6.

Step 1. Check $||\Phi|| < \epsilon$. If satisfied, the extrapolated dependent variables u are satisfactory, therefore go to step 4.

A polynomial extrapolator is maintained for u by the integration algorithm at little additional overhead. The term extrapolator is used rather than predictor, because the program does not maintain full error control on the variables u. Since this extrapolator may be of any order up to 12, the constraints $\Phi(u, v, t)$ will usually satisfy the above closure test, thus avoiding the following steps 2 and 3. In addition the partitioning of q into sets u and v depends upon q and t because Φ_q is a function of q and t. Since the extrapolator for u and the predictor for v have the same form, it is not necessary to interrupt the integration process to account for different sets u and v. Error control within the integration algorithm is maintained only on v since steps 1 to 3 are equivalent to maintaining error control on u.

Step 2. Evaluate Φ_q and perform L-U factorization to determine Φ_u, Φ_v, H and the partitioning of q into u and v.

Step 3. Iterate to determine u, using Equation 3.5, until $||\Phi|| < \epsilon$.

Step 4. Evaluate Φ_q, factor, and calculate $\dot{u}$ from Equations 3.16 and 3.17.

The partitioning of $\dot{q}$ into $\dot{u}$ and $s \equiv \dot{v}$ depends upon q and t, because Φ_q is a function of q and t. Since all elements of $\dot{q}$ are candidates for s, a polynomial extrapolator for $\dot{u}$ and a predicator for s are maintained by the numerical integration algorithm, with error control only on s. This facilitates smooth transition of elements of $\dot{q}$ into and out of s within the integration algorithm, with no interruption of the integration process. Error control within the integration algorithm is maintained only on s since Equations 3.16 and 3.17 are equivalent to maintaining error control on $\dot{u}$.

Step 5. Calculate $\dot{s}$ from Equation 4.11.

If required, calculate $\ddot{u}$ from Equations 3.22 and 3.23, and λ from Equation 3.30, where $\dot{q} = [\dot{u}^T, s^T]^T$ and $\ddot{v} = \dot{s}$. These calculations are not required for the integration steps that follow, however $\ddot{u}$ may be used to improve the extrapolator of $\dot{u}$ [28].

Step 6. Using the numerical integration PECE algorithm [28], predict q and $\dot{q}$ at the $(i + 1)^{st}$ time step. That is, predict v^{i+1} and s^{i+1} and extrapolate u^{i+1} and $\dot{u}^{i+1}$.

Step 7. Execute steps 1 to 5 to evaluate $\dot{u}$, $\dot{v}$, $\dot{s}$, and $\ddot{u}$.

Step 8. Using $\dot{u}$, $\dot{v}$, $\dot{s}$, and $\ddot{u}$ from step 7, correct u, v, s, and $\dot{u}$.

Correcting u and $\dot{u}$ is done only to improve the extrapolators of u and $\dot{u}$ for the next step which reduces predictor error if elements of u and $\dot{u}$ are subsequently picked up by v and s at the next step.

Step 9. Estimate integration error on v and s. Adjust the current time step and integration order to suit integration error requirements. If integration error is too large, go back to step 6, otherwise, step 10.

Step 10. Execute steps 1 to 5 again to obtain updated values for $\dot{q}$, $\ddot{q}$, and λ and report the solution if at or past the desired reporting times. Increment i and return to step 6 or stop, if the final time is reached.

4.5 Sparse Matrix Considerations

(a) Initial Conditions

As noted in Chapter 2 the nodal method of constraint formulation yields loosely coupled equations, with a corresponding sparse Jacobian matrix. If the system has a large number of constraints, sparse matrix manipulation algorithms will be efficient. The various equations in Chapters 3 and 4 can be expressed in different forms to avoid unnecessary calculation of sparse matrix products. The procedure for establishing initial conditions, similar to Section 4.3, is described in the following steps:

Step 1. Specify an initial estimate q^0 of the constrained system configuration. This estimate must be near the expected final configuration to insure convergence. If static equilibrium is to be obtained, the estimate must be near the desired equilibrium configuration to avoid convergence to alternate stable or unstable configurations, or divergence. Set $q = q^0$.

Step 2. If some elements of q are to remain fixed during initial assembly of the constrained system or static equilibrium, append to Equation 4.4, additional algebraic constraints of the form $q_j - q_j^0 = 0$, where j identifies the variables to be held fixed. Other types of algebraic relations between coordinates can also be formulated, as needed. These constraints will be removed prior to step 6.

Step 3. Evaluate $\Phi_q(q)$ (including constraints of step 2) and factor, as described in Appendix A, to determine a submatrix Φ_u of maximal rank and to determine a partitioning of q into u and v (dependent and independent sets, respectively). If Φ_q does not have full row rank then too many constraints or inconsistent constraints are specified. Therefore, some equations are dependent and they can be temporarily ignored in the iterative procedure of step 4. A matrix $\bar{\Phi}_q$ of rank 3n is constructed by appending the matrix $\bar{I}$ to Φ_q, as done in Section 3.5. The matrix $\bar{I}$ has 1's in columns corresponding to elements v of q. This matrix is then factored using sparse matrix algorithms.

Step 4. Iterate using Equation 3.36 to solve the independent constraint equations of step 2, until $||\bar{\Phi}|| < \epsilon$, where ϵ is a specified closure tolerance level. If there are dependent constraint equations, check them for violation. When dependent constraints are violated, inconsistent constraint conditions have been specified and a meaningless solution may result. If static equilibrium is not desired, go to step 6.

Step 5. Determine a new estimate for v that reduces $||R^v||$. From Equation 3.39 expressed as

$$\bar{\Phi}_q^T [\lambda_R v] = - Q$$

evaluate R^v and by numerical differencing evaluate dR^v/dv. Using the method of Appendix A, factor the above matrix dR^v/dv and solve the equation

$$\frac{dR^v}{dv} \cdot \Delta v = - R^v$$

for Δv. If $||R^v|| > \delta$, a specified force unbalance tolerance, return to step 3. If the iteration in steps 3 to 5 diverges, either inconsistent constraints have been imposed or initial estimates of system configuration are in error.

Step 6. Specify initial estimates for velocities $\dot{q}^0$ and set $\dot{q} = \dot{q}^0$. The velocities $\dot{q}$ are adjusted to satisfy Equation 3.13,

evaluated at q from the above steps. If some elements of $\dot{q}$ are to remain fixed at $\dot{q}^0$, append to Equation 3.13 additional equations of the form $\dot{q}_k - \dot{q}_k^0 = 0$, where k identifies the variables held fixed. Other constraint relations between velocities can be formulated as needed. These constraints will be removed prior to step 1 of transient analysis.

Step 7. Factor the modified matrix of Equation 3.13 as described in Appendix A to determine a partitioning of $\dot{q}$ into $\dot{u}$ and $\dot{v}$ (not necessarily the same partitioning as in step 3). Append to Equation 3.13 additional equations of the form $\dot{v} - \dot{v}^0 = 0$ to form $\bar{\Phi}_q \dot{q} + \bar{\Phi}_t = 0$. This is equivalent to appending a matrix $\bar{I}$ to Φ_q with 1's in columns corresponding to elements $\dot{v}$ of $\dot{q}$. Factor $\bar{\Phi}_q$ as in step 3. Calculate $\dot{u}$ using the above partitioning in Equation 3.39. If the above matrix lacks full row rank, some equations are dependent and must be checked for consistency. If dependent equations are not satisfied, inconsistent constraints on velocity are specified and a meaningless solution may result. Otherwise a consistent set of initial conditions is now available for transient analysis.

(b) Numerical Integration of the Equations of Motion

The steps in Section 4.4, modified to incorporate sparse matrix techniques, are as follows (See Section 4.4 for additional comments):

Let i, initially 0, be an indicator for the current time step; i.e., i = 0 implies $t = t_0$. Steps 1 to 5 are satisfied when starting from results of the preceding algorithm.

Step 1. Check $||\Phi|| < \epsilon$. If satisfied, the extrapolated dependent variables u are satisfactory, so go to step 4.

Step 2. Periodically evaluate Φ_q and perform L-U factorization to determine the partitioning of q into u and v. Append to Equation 4.4 additional equations of the form $v - v^p = 0$ to form $\bar{\Phi} = 0$ where v^p are the predicted independent variables. Factor the matrix $\bar{\Phi}_q$ using sparse matrix algorithms and go to step 3.

Step 3. Iterate to determine u, using Equation 3.36, until $||\bar{\Phi}|| < \epsilon$.

Step 4. Evaluate $\bar{\Phi}_q$, factor and calculate $\dot{q}$ from Equation 3.37. Noting that $\dot{q} = [\dot{u}^T, s^T]^T$, this step is equivalent to evaluating the differential equations $\dot{v} = s$.

Step 5. Combine Equations 2.19, 3.18, and 3.19 in matrix form as

$$\begin{bmatrix} M & \Phi_q^T \\ \Phi_q & 0 \end{bmatrix} \begin{bmatrix} \ddot{q} \\ -\lambda \end{bmatrix} = \begin{bmatrix} A + Q \\ - C \end{bmatrix}$$

This sparse matrix is factored, using sparse matrix algorithms, and the accelerations and Lagrange multipliers are determined. Since $\ddot{q} = [\ddot{u}^T, \dot{s}^T]^T$ this step is equivalent to evaluating the differential equations for $\dot{s}$.

Step 6. Using the numerical integration PECE algorithm, predict q and $\dot{q}$ at the $(i + 1)^{st}$ time step. That is, predict v^{i+1} and s^{i+1} and extrapolate u^{i+1} and $\dot{u}^{i+1}$.

Step 7. Execute steps 1 to 5 to evaluate $\dot{u}$, $\dot{v}$, $\dot{s}$, and $\ddot{u}$.

Step 8. Using $\dot{u}$, $\dot{v}$, $\dot{s}$, and $\ddot{u}$ from step 7, correct u, v, s, and $\dot{u}$.

Step 9. Estimate integration error on v and s. Adjust the current time step and integration order to suit integration error requirements. If integration error is too large, go back to step 6, otherwise, step 10.

Step 10. Execute steps 1 to 5 again to obtain updated values for $\dot{q}$, $\ddot{q}$, and λ, report the solution if at or past the desired reporting times. Increment i and return to step 6 or stop, if the final time is reached.

CHAPTER 5

PIECED INTERVAL ANALYSIS

5.1 Introduction

Dynamic systems characterized by one or more events of short duration, relative to total system response time, that have significant effect on the state are analyzed. Apart from these events, the system response is smooth. Numerical integration algorithms that descretize the time domain may miss events such as impulsive loads or may search inefficiently for abrupt state transition regions. As transition regions approach distinct event "logical times", discontinuities occur and the algorithms will fail to find a solution. The alternative is to sense logical times and handle each in the most efficient and accurate manner possible.

Pieced interval analysis implies division of the time domain into discrete intervals or stages. Stages may represent discontinuous logical events of zero time duration, such as momentum balance, continuous abrupt transition regions of short time duration, or long smooth regions. Each stage may require a different set of state equations, initial conditions, and analysis technique.

The points in time that separate stages are called logical times. They are determined by the occurance of well-defined events.

Since these logical times are usually functions of the system state, it may be impossible to specify them in advance. At best, one may write logical functions of the state that analytically define the occurance of important events. From these events the corresponding logical times are obtained. A logical events monitor built into the numerical integration algorithm is most effective in locating events and obtaining solutions at the logical times.

Associated with the logical events monitor, hence logical times, is user selectable programmed control logic that controls the analysis in the following stage(s). Actions taken depend upon the characteristics of each given problem and the type of analysis performed within each stage. Such actions may be as simple as imposing limits on integration time stepsize or relaxing error tolerances during short duration events, such as impulsive loading, or they may be as extensive as obtaining new initial conditions through momentum balance, redefining the system by adding or deleting bodies and constraints, selecting the most appropriate integration algorithm, and restarting the integration process.

An important application of pieced interval analysis arrises when dynamic analysis techniques are used in conjunction with design sensitivity analysis and optimal design algorithms. The basic optimal design problem with fixed starting and terminal end points, no state discontinuities, and no intermediate constraints is readily solved by existing techniques [34]. Design sensitivity analysis

and optimization of problems with discontinuities at logical times requires that one relax these restrictions and allow for variation in logical times and discontinuities that occur there. Such methods have been used for small scale problems [43]. The objective here is to extend and apply these methods to large scale systems with automated equation generation and solution.

5.2 Logical Events Monitor for Pieced Interval Analysis

Logical events, defining predictable points in time and state, are employed to facilitate efficient and orderly analysis of complex dynamic mechanical systems. Boundaries (marked by logical events) of successive stages in analysis often must be located before they are encountered in order to avoid numerical difficulties associated with the prediction of system state variables through abrupt or discontinuous transition regions. Therefore a "logical events monitor" is developed that utilizes the polynomial predictor of the numerical integration algorithm, to solve for logical times before predicting system state variables. Having identified one or more logical times, the monitor forces a solution precisely at the first such time, interrupts execution, and returns control to the user or user supplied subroutines for further action.

To define the boundaries of successive stages in the analysis of a mechanical system, a set of logical times t_i, $i = 1,\ldots, k'$, is defined by equations

$$\ell^j = \Omega^j(q, \dot{q}, t) = 0, \quad j = 1, \ldots, m' \tag{5.1}$$

The ordering of logical times is defined by the dynamical system state and time.

To utilize the polynomial predictor of the numerical integration algorithm, Equation 5.1 is differentiated with respect to time

$$\dot{\ell}^j = \Omega^j_q \dot{q} + \Omega^j_{\dot{q}} \ddot{q} + \Omega^j_t, \quad j = 1, \ldots, m' \tag{5.2}$$

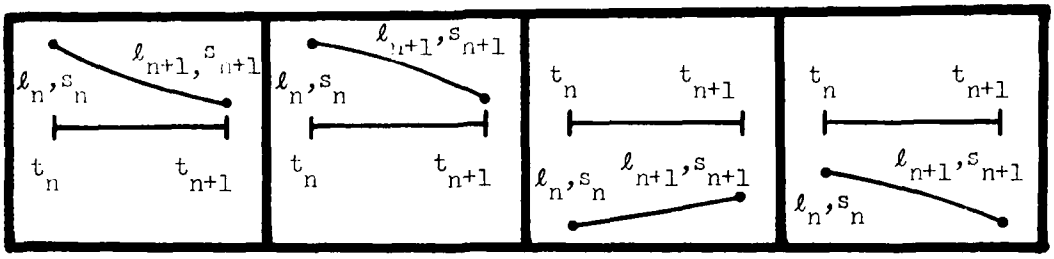
and integrated along with the system equations of motion. This allows freedom in the selection of integration algorithms, because the form of predictor is not involved in the formulation of Equation 5.2. Polynomial predictors use past history of state variables or their derivatives to extrapolate or predict them ahead in time. The logical variables ℓ^j are thus predicted ahead in time before the system state variables are predicted. If one or more logical variables should pass through zero, corresponding logical times have been passed. Each logical time is then found by solving for the zero of its corresponding polynomial predictor. The event corresponding to the smallest time step is thus chosen and becomes the point in time at which a solution is forced. If no logical variable passes through zero in a given time step, the integration algorithm proceeds as usual.

The success of this method relies on locating the zero crossing of each logical variable ℓ^j, and on determining a reduced stepsize

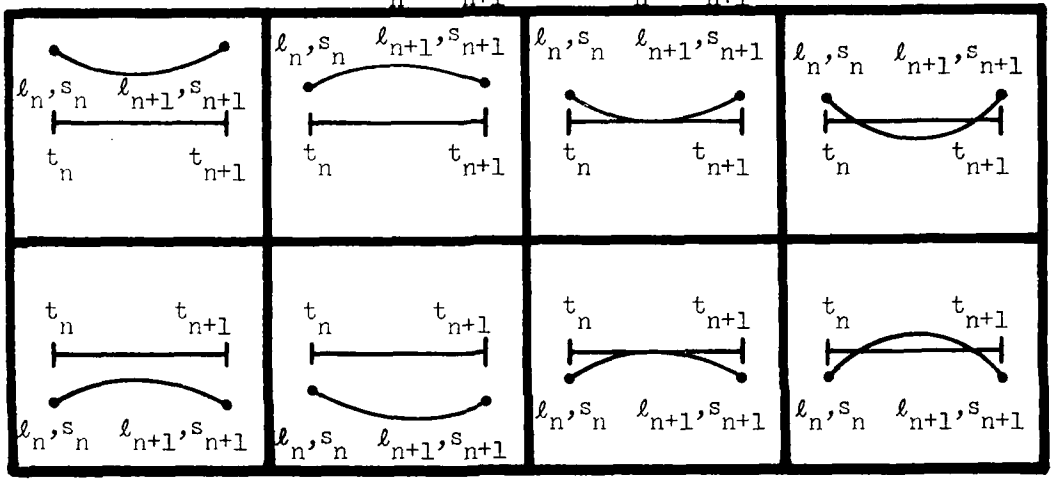
that forces a solution at the associated logical time. It is expected that logical functions are smooth enough so that no more than two zero crossings occur within any time step. It is possible, for example, when a logical function defines the starting and ending times of a very short impulse, that a logical function passes through zero twice within a single time step. Each zero crossing must be located in succession for the algorithms to work properly.

Presume that the integration algorithm has successfully reached time station t_n and the j^{th} logical variable has the value ℓ_n^j and time derivative s_n^j. The logical variable ℓ_n^j and slope s_n^j are predicted to time station t_{n+1} as ℓ_{n+1}^j and s_{n+1}^j. Figure 5.1 illustrates various possible combinations of ℓ^j and s^j. First, if $\ell_n^j \cdot \ell_{n+1}^j > 0$ and $s_n^j \cdot s_{n+1}^j > 0$ there are no zero crossings. Second, if $\ell_n^j \cdot \ell_{n+1}^j > 0$ and $s_n^j \cdot s_{n+1}^j < 0$, there are either no, one, or two zero crossings. Third, if $\ell_n^j \cdot \ell_{n+1}^j < 0$ and $s_n^j \cdot s_{n+1}^j$ is positive or negative there is one zero crossing. Case 1 requires no further action. Case 2 requires interval reduction until either case 1 or case 3 is achieved. Case 3 requires finding the zero crossing of ℓ^j and taking necessary action according to the programmed logic associated with that event.

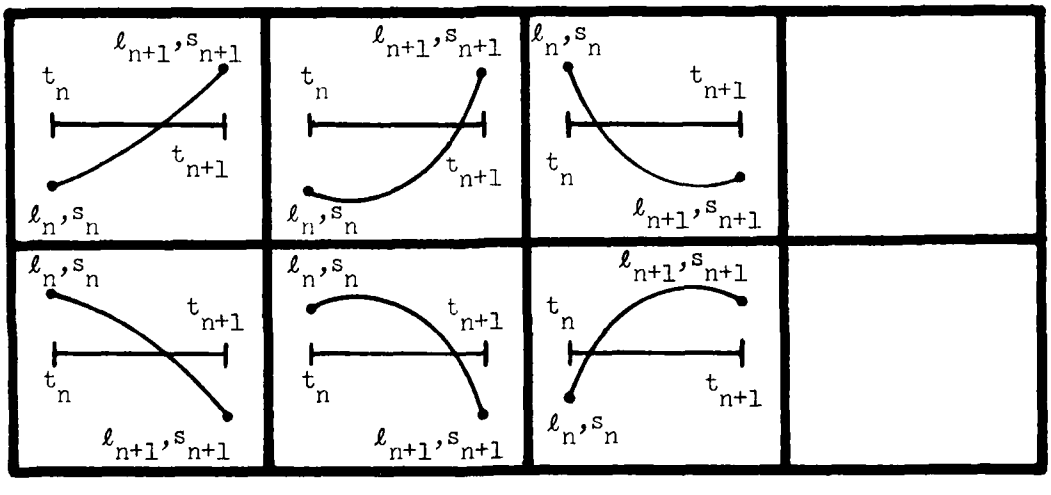
Finding the time step corresponding to the zero crossing of ℓ^j requires finding a root of a k^{th} order polynomial, where k is the order of the predictor employed in the numerical integration algorithm. The form of this polynomial depends on the integration algorithm's internal representation of the system state history. Two



Case I: $\ell_n \cdot \ell_{n+1} > 0$; $s_n \cdot s_{n+1} > 0$



Case II: $\ell_n \cdot \ell_{n+1} > 0$; $s_n \cdot s_{n+1} < 0$



Case III: $\ell_n \cdot \ell_{n+1} < 0$; $s_n \cdot s_{n+1}$ arbitrary

Figure 5.1. Various combinations of logical variable and its derivative at times t_n and t_{n+1}.

methods are employed in many predictor-corrector integration algorithms: 1) a Nordsieck vector representation, which stores k derivatives of the state variables at time station t_n [44], and 2) a divided difference table, based on the t_n^{th} and previous k values of the state variables [45]. Root finding techniques for each method are straight forward and will not be elaborated upon here.

5.3 Intermittant Motion in Constrained Systems

The phrases "discontinuous forces", "impulsive loads", and "discontinuous motion" appear often in the literature. In a micro-mechanics sense, such phenomena do not exist. However, to an observer who is interested in the dynamic response of systems over long periods of time, the idea of "discontinuous" or "impulsive" may be quite satisfactory. On the other hand, an observer interested in system dynamic response over short periods of time, specifically during significant events, will treat them in a continuous manner. Discontinuous and impulsive are thus labels that may be placed on continuous events, which in a macro-mechanics sense are effectively represented as discontinuous or impulsive in the observer's time frame. This illustrates the desirability of pieced interval analysis for certain applications, where entirely different mathematical models (and possibly analysis techniques) may be employed during different intervals of system motion.

In this section momentum balance equations are developed from Lagrange's equations of motion, assuming that discontinuous external forces or impulses are applied to a constrained system. The equations

are initially written for the case in which these events occur in a finite amount of time. It is assumed that the time interval (τ_1, τ_2) is small enough that the system configuration does not change appreciably, that is, $q(\tau_2) \approx (q(\tau_1)$. In addition, the constraints in Equation 3.1 are continuous functions of q and t.

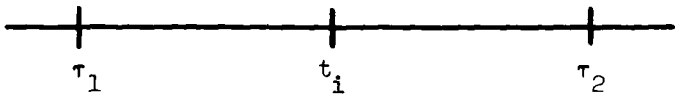


Figure 5.2. Event interval

Let t_i be a point in time at which a "violent event" occurs, which is to be approximated by a discontinuity. In reality, the event occurs over a time interval $\tau_1 < t_i < \tau_2$, as shown in Figure 5.2, and behavior is smooth except possibly at t_i. Integrate Equation 4.8 to obtain

$$\int_{\tau_1}^{\tau_2} [M^{vv} + H^T M^{uv} + (M^{vu} + H^T M^{uu})H]\dot{v}\, dt$$

$$= \int_{\tau_1}^{\tau_2} [A^v + Q^v + H^T(A^u + Q^u)$$

$$- (M^{vu} + H^T M^{uu})\ddot{u}^*]dt \qquad (5.3)$$

Since H is differentiable, integration by parts and using the mean value theorem gives

$$[M^{vv} + H^T M^{uv} + (M^{vu} + H^T M^{uu})H]\dot{v} \Big|_{\tau_1}^{\tau_2}$$

$$- \int_{\tau_1}^{\tau_2} \{ \frac{d}{dt} [M^{vv} + H^T M^{uv} + (M^{vu} + H^T M^{uu})H]\dot{v}$$

$$+ A^v + H^T A^u - (M^{vu} + H^T M^{uu})\ddot{u}^*\}dt$$

$$= \int_{\tau_1}^{\tau} Q^v \, dt + \hat{H}^T \int_{\tau_1}^{\tau_2} Q^u \, dt \qquad (5.4)$$

Where $\hat{H}$ is a matrix whose elements are those of H evaluated at points in (τ_1, τ_2).

Treating Q as impulsive at t_i, the integrals of generalized force are "generalized impulse",

$$P^v = \int_{\tau_1}^{\tau_2} Q^v dt \qquad \text{and} \qquad P^u = \int_{\tau_1}^{\tau_2} Q^u dt \quad .$$

Taking the limit in Equation 5.4 as $\tau_1 \to t_i$ and $\tau_2 \to t_i$, noting from Equations 3.19 and 3.22 that $\ddot{u}^*$ is bounded so $A^v + H^T A^u - (M^{vu} + H^T M^{uu})\ddot{u}^*$ is bounded, yields the "impulse momentum" equation at t_i as

$$[M^{vv} + H^T M^{uv} + (M^{vu} + H^T M^{uu})H][\dot{v}(t_i+) - \dot{v}(t_i-)]$$

$$= P^v + H^T P^u \tag{5.5}$$

This prescribes the velocity jump in v due to impulsively applied loads.

It is important to note that Equation 5.5 involves impulse and momentum of all elements of the mechanical system. This is crucial, since the bodies making up the system interact through constraints, so an impulse-momentum balance relation involving only the bodies on which the impulsive force acts is impossible. Deriving the relation of Equation 5.5 by manual calculation would be extremely difficult and time consuming. One of the strongest points of the method presented here is the automatic assembly of the coefficient matrices of Equation 5.5.

Figure 5.3. Impacting bodies

For impact of bodies i and j, as shown in Figure 5.3, a coefficient of restitution e provides the relative velocity relation in direction $\bar{n}$ (where $\bar{n}$ is taken as a row vector) as

$$\bar{n}\left[\dot{q}^{i}(t_i+) - \dot{q}^{j}(t_i+)\right] = -e\bar{n}\left[\dot{q}^{i}(t_i-) - \dot{q}^{j}(t_i-)\right] \qquad (5.6)$$

or with $N^T \in R^{3n}$, $N = \left[N^u \; , \; N^v\right]$. Equation 5.6 may be written

$$N^v \dot{v}(t_i+) + N^u \dot{u}(t_i+) = -e\left[N^v \dot{v}(t_i-) + N^u \dot{u}(t_i-)\right]$$

or using Equation 3.17, this is

$$\left[N^v + N^u H\right]\dot{v}(t_i+) + N^u \dot{u}^*(t_i+)$$

$$= -e\left[N^v + N^u H\right]\dot{v}(t_i-) - e N^u \dot{u}^*(t_i-) \qquad (5.7)$$

Equation 3.16 implies that $\dot{u}^*$ is a function only of q and t and constraints are continuously differentiable in q and t. Therefore, $\dot{u}^*(t_i+) = \dot{u}^*(t_i-)$ and Equation 5.7 reduces to

$$\left[N^v + N^u H\right]\dot{v}(t_i+) = -e\left[N^v + N^u H\right]\dot{v}(t_i-)$$

$$-(e+1)N^u \dot{u}^*(t_i-) \qquad (5.8)$$

The generalized impulse of the force f(t) in Figure 5.3 is

$$P = \int_{t_i-\epsilon}^{t_i+\epsilon} f(t)N^T dt = pN^T \tag{5.9}$$

where

$$p = \int_{t_i-\epsilon}^{t_i+\epsilon} f(t)dt \tag{5.10}$$

Defining the partitioning $P^v = pN^{v^T}$ and $P^u = pN^{u^T}$, Equation 5.3 gives

$$[M^{vv} + H^T M^{uv} + (M^{vu} + H^T M^{uu})H][\dot{v}(t_i+) - \dot{v}(t_i-)]$$

$$= p\,[N^{v^T} + H^T N^{u^T}] \tag{5.11}$$

The above equations are put into matrix form for solution as follows. Subtract the identity

$$[N^v + N^u H]\,\dot{v}(t_i-) = [N^v + N^u H]\,\dot{v}(t_i-)$$

from Equation 5.8 and define

$$\Delta\dot{v}_i = \dot{v}(t_i+) - \dot{v}(t_i-) \tag{5.12}$$

The matrix equation thus becomes

$$\begin{bmatrix} M^{vv} + H^T M^{uv} + (M^{vu} + H^T M^{uu})H & (N^{v^T} + H^T N^{u^T}) \\ (N^v + N^u H) & 0 \end{bmatrix} \cdot \begin{bmatrix} \Delta\dot{v}_i \\ -p \end{bmatrix}$$

$$= \begin{bmatrix} 0 \\ -(e+1)\{(N^v + N^u H)\dot{v}(t_i-) + N^u \dot{u}^*(t_i-)\} \end{bmatrix} \qquad (5.13)$$

whose solution yields the desired velocity jumps and magnitude of impulse at the body surfaces.

5.4 Pieced Interval Computational Algorithm

The development in this and previous chapters is combined into a general pieced interval analysis algorithm. In the discussion to follow, for notational convenience, assume that logical variable ℓ^j identifies t_i, the end of the ith stage. Associated with ℓ^j is a set of system state equations for the $(i+1)^{st}$ stage (which may be dynamic equations of motion or momentum balance equations). In addition, a different set of constraint equations Φ^{i+1} (which may include boundary constraints on velocity) may be used at this stage. A different set of logical functions Ω^{i+1}, compatible with the above constraint and state equations, may also be introduced.

Prior to implementing the algorithm, it is assumed that all possible system events have been anticipated and appropriate logical event predictors and corresponding constraint and state equations have been formulated. Knowledge of the sequence of events is unnecessary, unless this knowledge can be used to reduce the complexity of equation formulation. Further, it is assumed that no event takes place prior to achieving the initial static equilibrium configuration or appropriate initial conditions at time $t = t_0$.

The computational algorithm is as follows:

Step 1. Obtain initial conditions consistent with constraints, using the algorithm of Section 4.3.

Step 2. Select the appropriate set of state equations, constraints, logical functions, etc., based on the current active logical events monitor flag or continue with previous equations.

Step 3. If a momentum balance is required, solve for velocity jumps as in Section 5.3 and return to Step 2.

Step 4. Using the techniques described in Section 5.2 check for active logical variables (one or more ℓ^j passing through zero) in the next time step to be attempted by the numerical integration algorithm. Solve for time steps corresponding to the zeros of polynomial predictors of active ℓ^j (if any), set the integration time step to the smallest step (first occurring ℓ^j) and set the corresponding logical events monitor flag. Advance the solution to this point in time by executing Step 5.

Step 5. Perform dynamic analysis using the algorithm described in Section 4.4. If the solution is advanced by the above time step return to Step 2, unless the end of the simulation interval is reached, in which case stop. If a smaller time step is taken to meet integration error requirements, reset the logical events monitor flag (since the event has not yet been reached) and return to Step 2.

5.5 Sparse Matrix Considerations

To maintain program efficiency when mechanical systems become large, sparse matrix algorithms should be employed and matrix products avoided. The methods of sparse matrix computations described in previous chapters will replace the corresponding full matrix operations in the algorithm of Section 5.4. It is possible to show that Equation 5.13 can be expressed in a form suitable for sparse matrix computations by first writing it in permuted form as

$$\begin{bmatrix} M^{uu} & M^{uv} & \Phi_u^T & N^{u^T} \\ M^{vu} & M^{vv} & \Phi_v^T & N^{v^T} \\ \Phi_u & \Phi_v & 0 & 0 \\ N^u & N^v & 0 & 0 \end{bmatrix} \begin{bmatrix} \Delta\dot{u}_i \\ \Delta\dot{v}_i \\ -\tilde{\lambda} \\ -p \end{bmatrix}$$

$$= \begin{bmatrix} 0 \\ 0 \\ 0 \\ -\{e + 1\}\{N^u \dot{u}(t_i-) + N^v \dot{v}(t_i-)\} \end{bmatrix} \tag{5.14}$$

where $\tilde{\lambda}$ is a vector of Lagrange multipliers. This equation is equivalent to the unpermuted matrix equation

$$\begin{bmatrix} M & \Phi_q^T & N^T \\ \Phi_q & 0 & 0 \\ N & 0 & 0 \end{bmatrix} \begin{bmatrix} \Delta\dot{q}_i \\ -\tilde{\lambda} \\ -p \end{bmatrix}$$

$$= \begin{bmatrix} 0 \\ 0 \\ -\{e + 1\}\{N \dot{q}(t_i-)\} \end{bmatrix} \tag{5.15}$$

To show the equivalence between Equation 5.13 and 5.14 carry out the matrix products in Equation 5.14

$$M^{uu} \Delta\dot{u}_i + M^{uv} \Delta\dot{v}_i - \Phi_u^T \tilde{\lambda} - N^{u^T} p = 0 \tag{5.16}$$

$$M^{vu}\,\Delta\dot{u}_i + M^{vv}\,\Delta\dot{v}_i - \Phi_v{}^T\,\tilde{\lambda} - N^{v^T} p = 0 \qquad (5.17)$$

$$\Phi_u\,\Delta\dot{u}_i + \Phi_v\,\Delta\dot{v}_i = 0 \qquad (5.18)$$

$$N^u\,\Delta\dot{u}_i + N^v\,\Delta\dot{v}_i = -\{e+1\}\{N^u\,\dot{u}(t_i-) + N^v\,\dot{v}(t_i-)\} \qquad (5.19)$$

Since

$$\Phi_u\,\dot{u}_i(t_i-) + \Phi_v\,\dot{v}_i(t_i-) + \Phi_t(t_i-) = 0 \quad ,$$

$$\Phi_u\,\dot{u}_i\,(t_i+) + \Phi_v\dot{v}_i(t_i+) + \Phi_t(t_i+) = 0 \quad ,$$

and

$$\Phi_t(t_i-) = \Phi_t(t_i+)$$

then

$$\Phi_u\{\dot{u}_i(t_i+) - \dot{u}_i(t_i-)\} + \Phi_v\{\dot{v}_i(t_i+) - \dot{v}_i(t_i-)\} = 0$$

and Equation 5.18 is satisfied. Equation 5.18 also implies that

$$\Delta\dot{u}_i = -\,\Phi_u{}^{-1}\,\Phi_v\,\Delta\dot{v}_i = H\,\Delta\dot{v}_i \qquad (5.20)$$

Equation 5.19 is simply the second equation of Equation 5.13 where Equations 3.17 and 5.20 are employed.

Since Φ_u^T is nonsingular in Equation 5.16, solve for $\tilde{\lambda}$ and substitute into Equation 5.17, thus

$$M^{vu}\,\Delta\dot{u}_i + M^{vv}\,\Delta\dot{v}_i + \Phi_v^T\,\Phi_u^{-T}(-M^{uu}\,\Delta\dot{u}_i - M^{uv}\,\Delta\dot{v}_i + N^{u^T}p) - N^{v^T}p = 0$$

Combining terms and using Equation 5.20 yields

$$\{M^{vv} + H^T M^{uv} + (M^{vu} + H^T M^{uu})H\}\,\Delta\dot{v}_i - (N^{v^T} + H^T N^{u^T})p = 0$$

which is the first equation of Equation 5.13. Equation 5.15 is thus solved to determine changes in velocity due to impact between bodies.

CHAPTER 6

NUMERICAL EXAMPLES

6.1 Introduction

Numerical examples are presented to demonstrate generality and improved program efficiency, resulting from the analytical methods developed in previous chapters. Generality is demonstrated by the ease of representing models of complex dynamic mechanical systems. Efficiency is demonstrated by contrasting computer simulation times for the various solution methods.

The analytical methods developed in Chapters 3 and 4 demonstrate the greatest improvement in program efficiency when mechanical systems are heavily constrained. The integration algorithm of Chapter 2 iteratively solves $(6n + r)$ differential and algebraic equations, whereas the algorithm of Chapter 4 solves $6n - 2r$ differential equations; n being the number of system rigid bodies and r the number of independent algebraic constraint equations. Defining system degree of freedom as

$$dof = 3n - r$$

the ratio of number of equations for the two methods is

$$(6n - 2r)/(6n + r) = (2\ dof)/(9n - dof)$$

For systems with many bodies and a few degrees of freedom, this ratio is small.

The three degree-of-freedom link gear multiplier consisting of twelve bodies, shown in Figure 6.1, has an equation ratio of 0.057. The one degree-of-freedom Peaucellier Lipkin mechanism in Figure 6.2 consists of 8 bodies and has an equation ratio of 0.028. A significant reduction in matrix and numerical integration operations is expected. Increased efficiency also results from the use of single precision computer arithmetic, reduced number of corrector iterations, and larger time step sizes.

Operations count for solution of sparse matrix problems is of the order $N^{1.6}$, where N is the matrix dimension [46]. The ratio $[(3n)/(9n - dof)]^{1.6}$ is a rough estimate of the reduction in operations count. When $dof \ll 3n$ this ratio approaches $3^{(-1.6)} = 0.17$. As $dof \to 3n$ the modified constraint matrix (see Equation 3.36) corresponding to the numerator of this ratio approaches an identity matrix, requiring substantially less than $(3n)^{1.6}$ operations. More than a 10 to 1 reduction in simulation time for the above mechanical system simulations was observed.

As the equation ratio increases, one expects less gain in program efficiency. For example, the mechanical system described in the next section, consisting of 12 bodies with 24 degrees of freedom, has an equation ratio of .596 and a corresponding 4 to 1

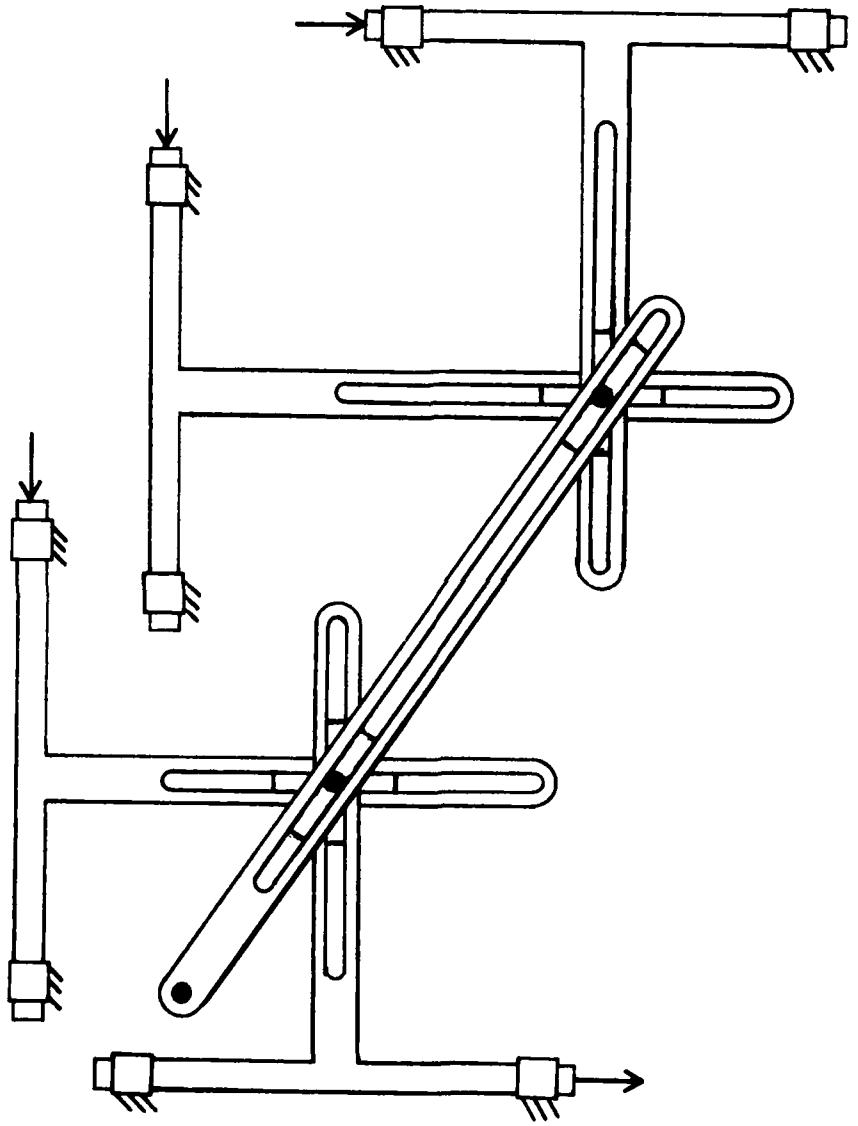


Figure 6.1. Link gear multiplier.

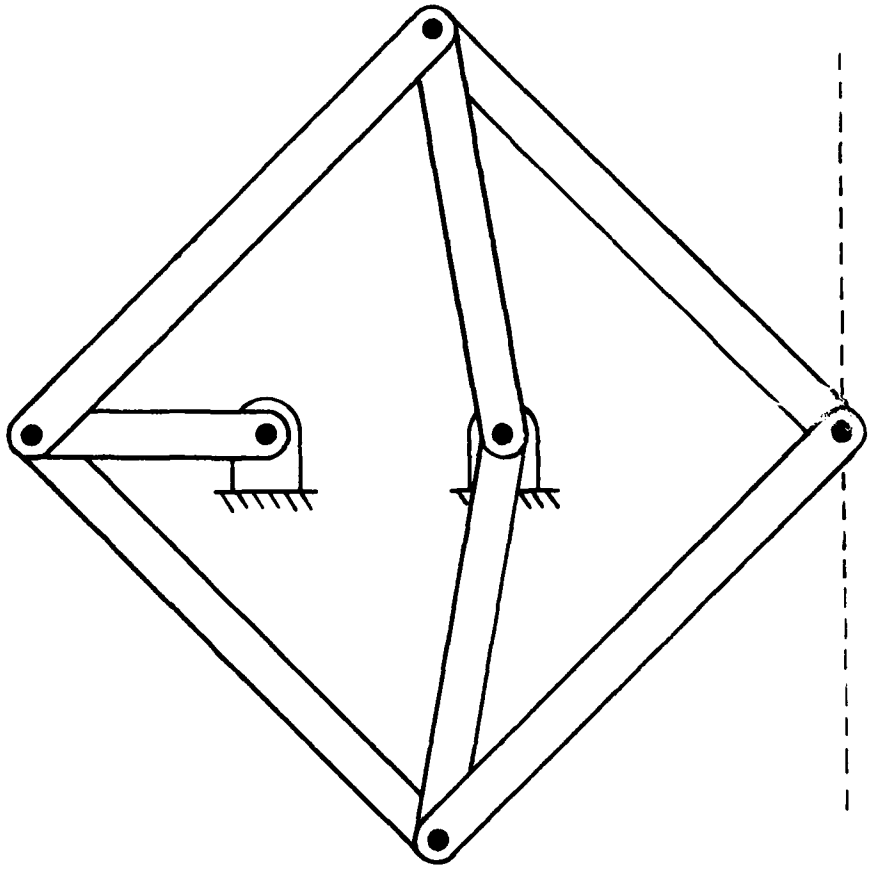


Figure 6.2. Peaucellier Lipkin mechanism.

reduction in execution time. Note that the above equation ratio for both methods includes 5 additional differential equations that describe control system dynamics. Another mechanical system (not presented in this report) consisting of 8 bodies and 13 degrees of freedom, has an equation ratio of .44 and a corresponding 7 to 1 reduction in execution time [21]. Matrix operation counts also play important roles in reduction of computer time.

6.2 Two Articulated Army M-113 Armored Personnel Carriers

(a) Vehicle Pitch Position and Force Feedback Control

A research test vehicle shown in Figure 6.3 was built by the Army for the purpose of investigating off-road performance of articulated tracked vehicles [47, 48]. This vehicle is selected for simulation here, because most of its parameters are known, some field test data is available for comparison purposes, and its pitch articulation (while negotiating terrain and obstacles) can be adequately represented by a planar model.

The test vehicle consists of two identical M-113 Army armored personnel carriers, coupled together by a spherical ball socket that allows relative vehicle pitch, yaw, and roll. This connection is located close to the rear of the front vehicle, to give the rear vehicle an increased moment arm, thus allowing it to remain nearly horizontal as the front vehicle is elevated. This allows the articulated vehicle to negotiate step obstacles up to 5 feet in height.

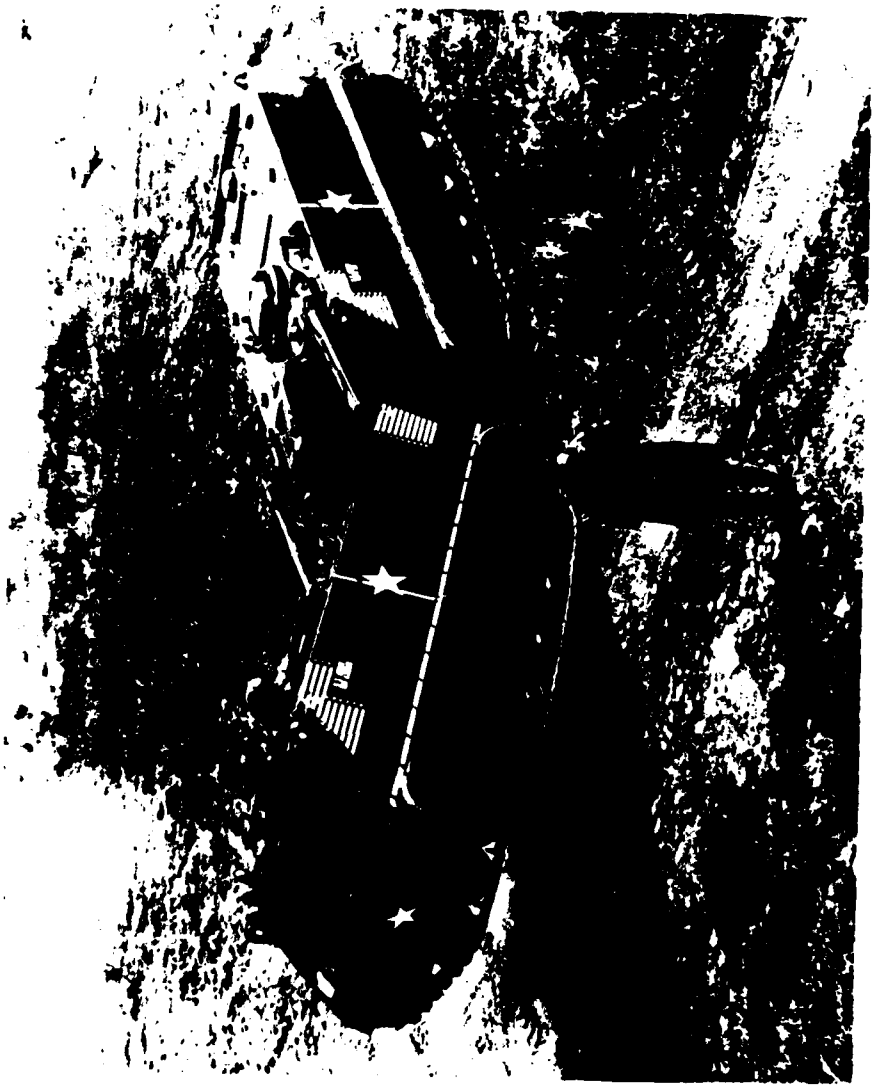


Figure 6.3. Coupled M113's.

Relative pitch and yaw motion of the system is controlled by two hydraulic cylinders acting between the vehicles as shown in Figure 6.3. Yaw motion occurs when one cylinder extends and the other simultaneously contracts. Yaw will not be considered in this investigation. Therefore, equal actuator lengths, thus equal actuator extension and contraction rates, will be assumed. The hydraulic control system is capable of maintaining equal actuator extension rates, while negotiating various obstacles.

Relative pitch and yaw can be controlled by the vehicle operator, via a control stick. If the stick is pushed to the right or left, the front vehicle yaws to the right or left, respectively. If the stick is pulled back or pushed forward, the front vehicle pitches up or down, respectively. Combinations of stick movements yield general yaw and pitch motions. It is assumed that only pitch command signals are given in the following planar simulations.

For the operator to relate control stick displacement to relative vehicular pitch displacement, a proportional control is required. That is, pulling the stick halfway back causes the vehicle to pitch up to 50% of its maximum allowed displacement, etc. Proportional control is provided through positional feedback to the control system by monitoring the hydraulic actuator extension or contraction. Proportional yaw control is obtained by a similar process.

Typical operator actions for negotiating a terrain obstacle might be as follows: To negotiate an obstacle pull the stick back

far enough such that the front unit can mount the obstacle. Gradually drive forward while adjusting stick position so that the vehicle maintains maximum conformity to the terrain and obstacle. This technique generally gives the smoothest ride and better vehicle traction and control. However, the operator must continually observe the vehicle's position relative to terrain, in order to make these adjustments. This requirement may limit his ability to perform other tasks.

Maximum terrain conformity is achieved if the hydraulic actuators are removed from the vehicles. Since this isn't possible, the alternative is to maintain zero pressure within the actuators. This is accomplished by providing force feedback from the hydraulic actuators to the operator, via the control stick.

Suppose the operator pulls back on the control stick and the vehicle begins to pitch up. Pressure builds up in the rod end of the cylinder and a voltage from a pressure transducer is generated. This signal is amplified and applied to an electro-mechanical device that pulls forward on the control stick, thus increasing its resistance to the operator's applied force. This gives him a feel for the pressure within the hydraulic actuator. He has simply to move the control stick in the direction it wants to go. In fact, he may remove his hand and the control stick will automatically position itself for minimum pressure and best terrain conformity. Therefore, force feedback allows the operator to devote more of his attention to other duties.

The control system cannot respond to system pressure changes instantaneously, because of inertia and hydraulic system compliance. Thus, at best, it can only maintain a zero average pressure over some time interval when the vehicles experience rapid relative pitch displacements. In fact, pressure transients may become quite high, as evidenced in results of simulations. A complete description of the electro-mechanical-hydraulic control system model is given, following a description of the vehicular system model.

(b) Elements of the Articulated Vehicle Mechanical Model

The primary components of an M113 APC are the chassis and ten roadwheels, five per side. The roadwheels are connected to the chassis by roadarms that are rigidly attached to torsion bars that provide chassis suspension. Each vehicle has two front drive and two rear idler sprockets. Segmented tracks, as shown in Figure 6.3, pass beneath the roadwheels and around the sprockets to form continuous loops. Power is transferred from the engine to drive sprocket and then to track that propels the vehicle. The roadwheels simply roll along the inner track surfaces. Shock absorbers provide damping in the front and rear suspensions.

A representative planar model is established from the above description. It is cost prohibitive to establish a model that minutely simulates the actual system. Therefore, various simplifications can be made where minor effects are ignored. Since motion in a plane is considered, model symmetry is employed to reduce

problem size. It is supposed that opposing roadwheels experience equal displacements, hence can be combined into single bodies having double mass and rotational inertia. Suspension stiffness and damping, track properties, etc., are doubled.

Track dynamics is not being studied here, so the mass and rotational inertia of the track segments beneath roadwheels is equally distributed among them. That part of the track supported by the chassis is lumped with the chassis mass and moment of inertia. Track supportive effects on chassis and roadwheels are significant and will be included in the model.

An outline drawing of the model is shown in Figure 6.4. This model consists of 12 bodies, two chassis (B11 and B12) and ten roadwheel pairs (B1 to B10). Roadarms, represented by massless links described in Chapter 2, connect roadwheels to the chassis. To simplify a track model, roadwheels are free to rotate, but only small rotations about initial angular positions are considered. Large rotational displacement of roadwheels is prevented by the track model.

Vertical force versus vertical roadwheel-displacement-suspension curves and vertical force versus vertical roadwheel-velocity-damping curves were determined experimentally (Figures 6.5 and 6.6). These forces are incorporated into the model employing features of the standard spring-damper-actuator elements described in Chapter 2. Elements S1 to S10, connected between chassis and roadwheel centers, are given arbitrarily long lengths; i.e., 1000 inches so that their forces relative to chassis remain essentially vertical for all

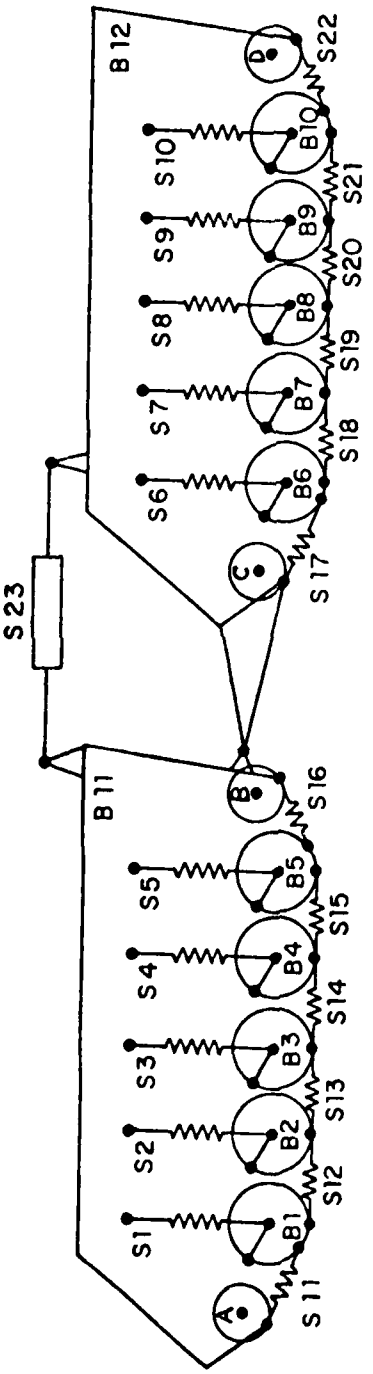


Figure 6.4. A computer model of the coupled M113's.

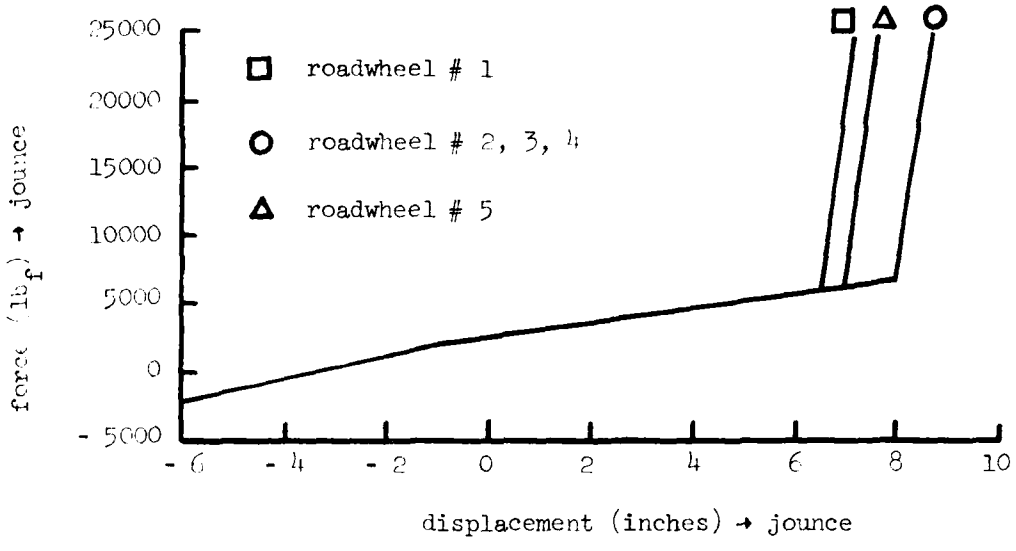


Figure 6.5. Torsion bar suspension characteristics

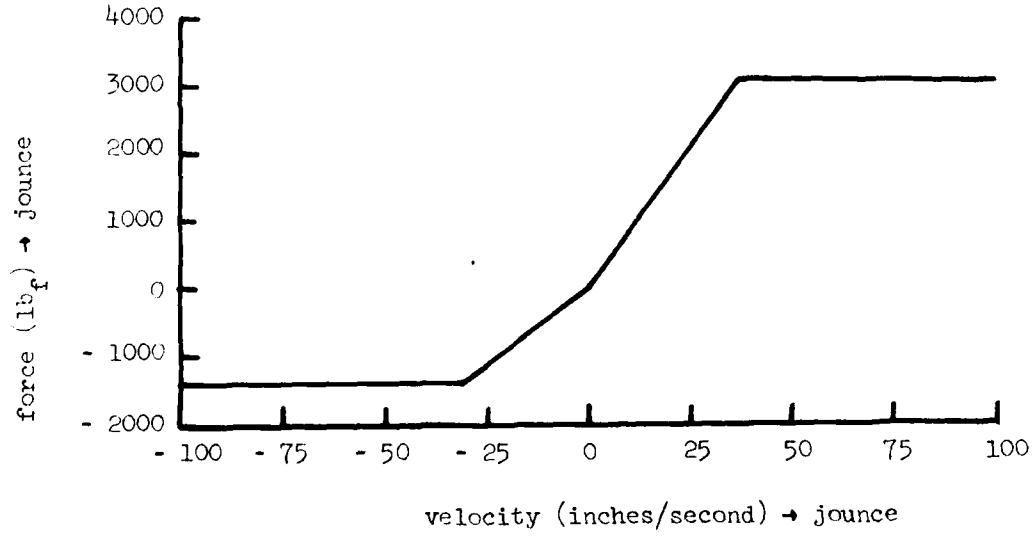


Figure 6.6. Suspension damping characteristics

roadwheel positions. Vertical roadwheel displacement relative to chassis is given by $(\ell_{O_{ij}} - \ell_{ij})$, where ℓ_{ij} is the current element length and $\ell_{O_{ij}}$ is a constant reference element length. Vertical roadwheel velocity, relative to the chassis, is given by $(-\dot{\ell}_{ij})$. These standard variables are available to user supplied subroutines in the program. Taking k_{ij} and c_{ij} as zero in Equation 2.9 yields an actuator element force of

$$\vec{F}_{ij} = F_{O_{ij}}(\vec{R}_{s_{ij}}/\ell_{ij})$$

Suspension and damping force functions $F_s(\ell_{O_{ij}} - \ell_{ij})$ and $F_d(-\dot{\ell}_{ij})$ are supplied by cubic spline curve fits [49, 50] to the data (straight line approximations have been successfully employed as well) [51]. The actuator force, evaluated as

$$F_{O_{ij}} = F_s(\ell_{O_{ij}} - \ell_{ij}) + F_d(-\dot{\ell}_{ij})$$

is automatically added to the system generalized forces.

(c) Elements of the Track and Terrain Interaction Model

Track forces on the chassis and roadwheels are included through spring-damper elements S11 to S22 (Figure 6.4). It is assumed that roadwheels remain in contact with the track throughout the simulation. Therefore, elements are connected directly to the bottom of the roadwheels. Elements S11, S16, S17, and S22 are also connected

directly to the chassis at the approximate points of track tangency to the drive and idler sprockets. These elements provide track interaction forces between the chassis and roadwheels. Each element segment has an effective stiffness determined by its approximate length and a track stiffness per unit length. Internal track damping is included to account for rubber bushing, rubber track pad, and ground dissipation. These coefficients depend nonlinearly on track displacement, since track segments cannot support compression.

Tracks are given a pretension so they will remain tight in various dynamic situations. Each element S11 to S22 is thus given an initial pretension of 10000 pounds. In order to determine changes in track tension (from equilibrium preload levels), due to dynamic loading, it is necessary to have reference track lengths $\ell_{O_{ij}} = \ell_{ij}$ at equilibrium for each of the above elements. These lengths cannot be determined when the model is established, because the equilibrium configuration is unknown. They can be evaluated directly by the program as equilibrium is established as follows: Set $k_{ij} = c_{ij} = 0$ and $F_{O_{ij}} = 10000\ lb_f$ in Equation 2.9 for each element S11 to S22. Constrain angular rotation of each roadwheel, using initial condition flags on input data cards to keep element attachment points at the bottom of the wheels. Solve for equilibrium and set each element $\ell_{O_{ij}}$ to ℓ_{ij}, which then becomes the reference length for dynamic analysis. Equation 2.9 for the dynamic track model then becomes

$$\bar{F}_{ij} = [k_{ij}(\ell_{ij})(\ell_{ij} - \ell_{O_{ij}}) + c_{ij}(\ell_{ij})\dot{\ell}_{ij} + F_{O_{ij}}][\vec{R}_{s_{ij}}/\ell_{ij}]$$

where $k_{ij}(\ell_{ij})$ and $c_{ij}(\ell_{ij})$ are determined to prevent the coefficient of $[\vec{R}_{s_{ij}}/\ell_{ij}]$ from becoming negative.

Roadwheels must be prevented from penetrating the terrain surface. That is, penetration should be no greater than that achieved by local wheel, track, and ground deformation. Support generally occurs at one or more points on the lower arc of a wheel, due to direct terrain contact or indirect track support. Support reaction forces may not act vertically. The general model for arbitrary terrain is rather complex. For the current discussion, assume a level terrain (a more representative model for irregular terrain is presented later), hence roadwheel support reaction forces remain vertical. These forces may then be moved to the wheel centers without changing the model. Since roadwheels are circular, the terrain reference elevation can be shifted up to the wheel centers for determination of relative wheel-terrain elevation. Global location of the i^{th} roadwheel center is given by coordinates x_i and y_i (see Chapter 2). If g_i is global terrain elevation and r_w is roadwheel radius, then wheel penetration into level terrain is

$$p_i = (g_i + r_w - y_i)$$

If wheel reaction force is developed according to the curve in Figure 6.7, the force $F_w(p_i)$ is then added to Q_y^i (see Equation 2.10). If $p_i \leq 0$, $F_w(p_i) = 0$ and the i^{th} wheel is not touching the terrain. A damping force $D_w(p_i, \dot{p}_i)$, with $\dot{p}_i = \dot{g}_i - \dot{y}_i$, is also added to

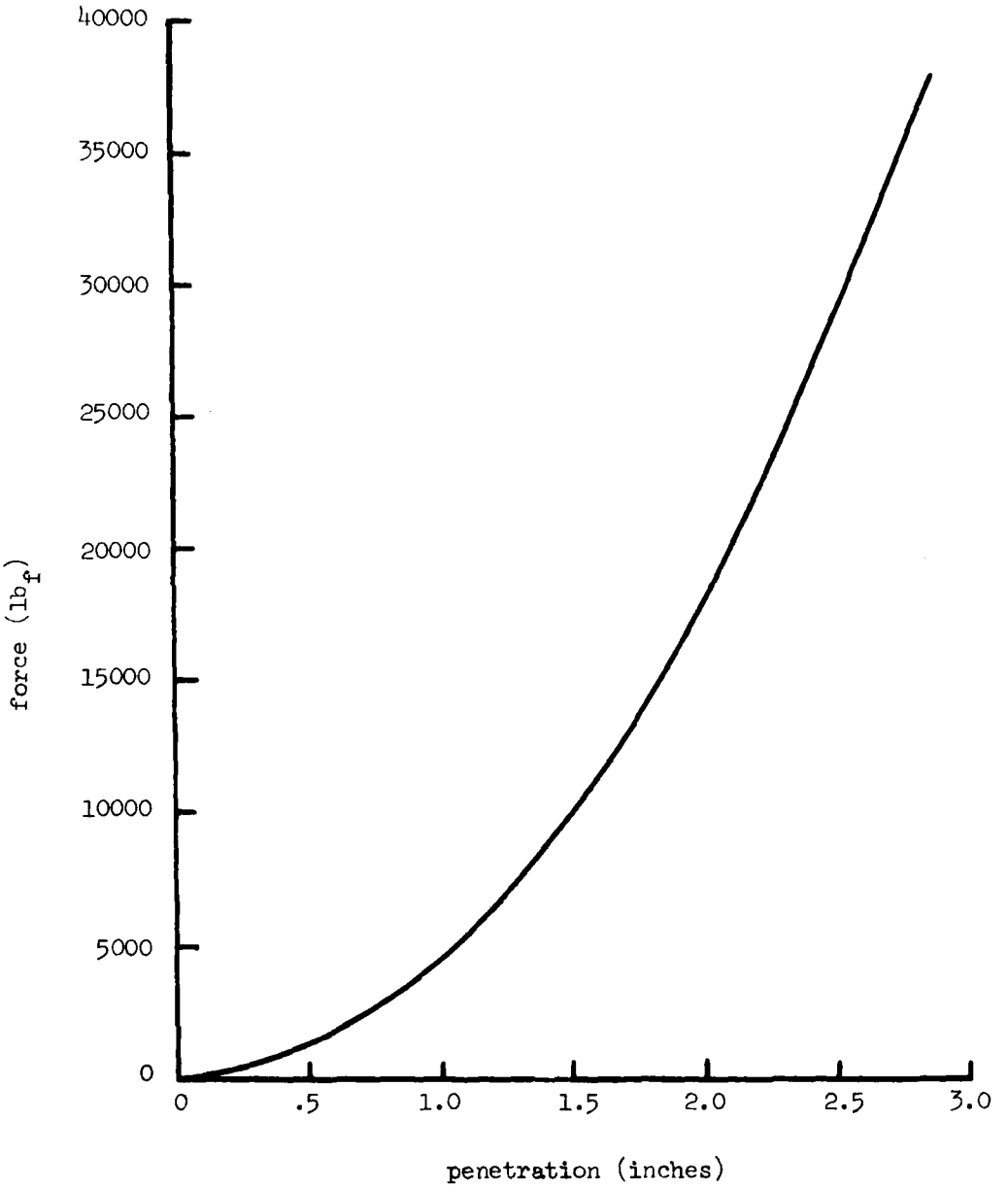


Figure 6.7. Roadwheel-ground force--penetration curve.

Q_y^i to include roadwheel rubber-, track rubber-, and ground-damping effects. Damping force D_w reduces to zero as p_i decreases to zero.

When vehicle pitch displacements are large, other points on the chassis may contact ground. The most likely places are the drive and idler sprockets, labeled A to D in Figure 6.4. Sprocket centers are located by global points x_{p_j} and y_{p_j}. With sprocket radii r_{s_j}, one may obtain support and damping forces as above. These forces are then included in generalized forces Q_y^i and Q_φ^i, for the proper bodies.

As noted in earlier discussion, the usual procedure for representing wheel-terrain interference is to monitor the distance between the bottom point on the wheel and the point on the terrain directly below it. If, as in Figure 6.8, the terrain slopes or is irregular, other points on the wheel may make first contact. Thus, the wheel will be at an incorrect height and the support force on the wheel will act incorrectly. A modified wheel trajectory may be obtained by moving a rigid wheel of a given radius, say that of the roadwheel, along the terrain surface to obtain a trajectory of the lowest point on the wheel [52]. The wheel trajectory for the terrain in Figure 6.8 is shown in Figure 6.9. As the lowest point on a wheel follows this trajectory, there will always be some point on the wheel just touching the terrain, but there will be no terrain penetration. This is illustrated in Figure 6.10. For the purpose of this simulation, it is sufficient to assume that the resultant support force vector acts along a line passing through the point of

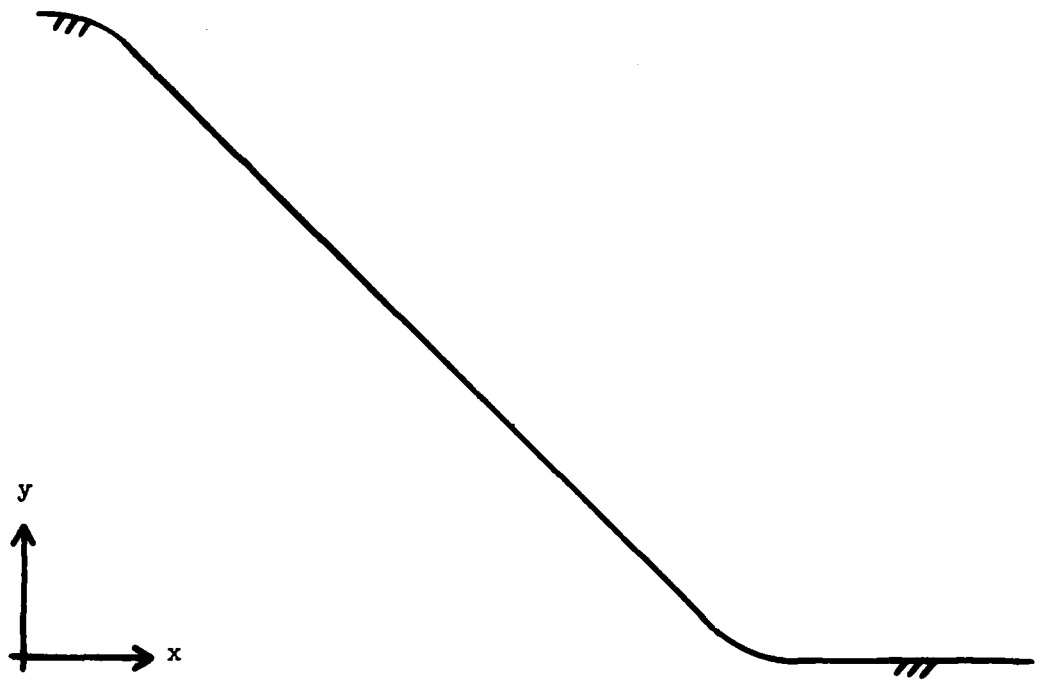


Figure 6.8. Terrain contour of 36 inch step obstacle.

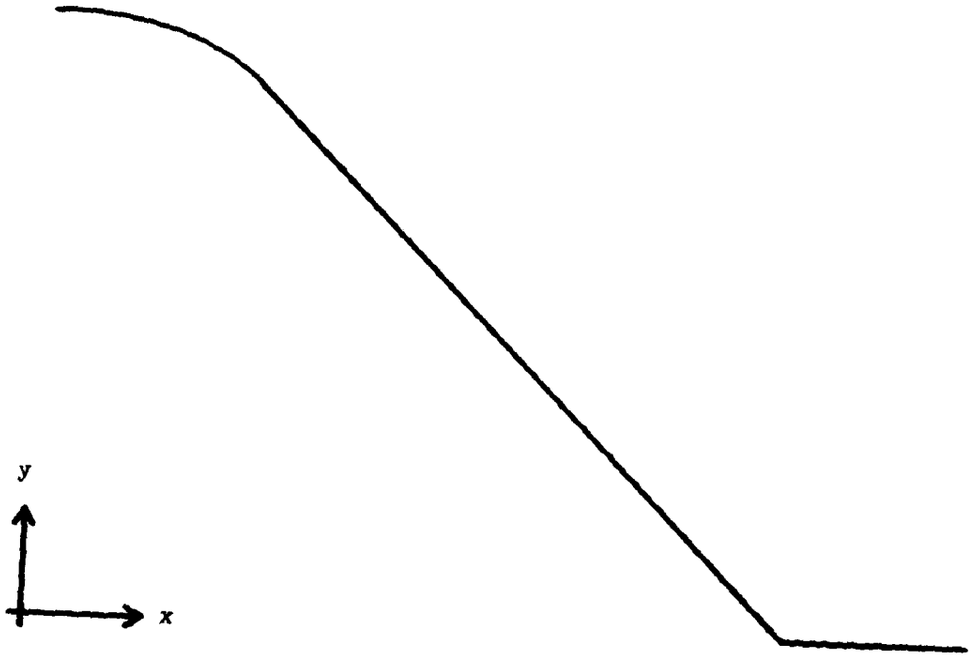


Figure 6.9. Trajectory of 12 inch wheel rolled over a 36 inch step obstacle.

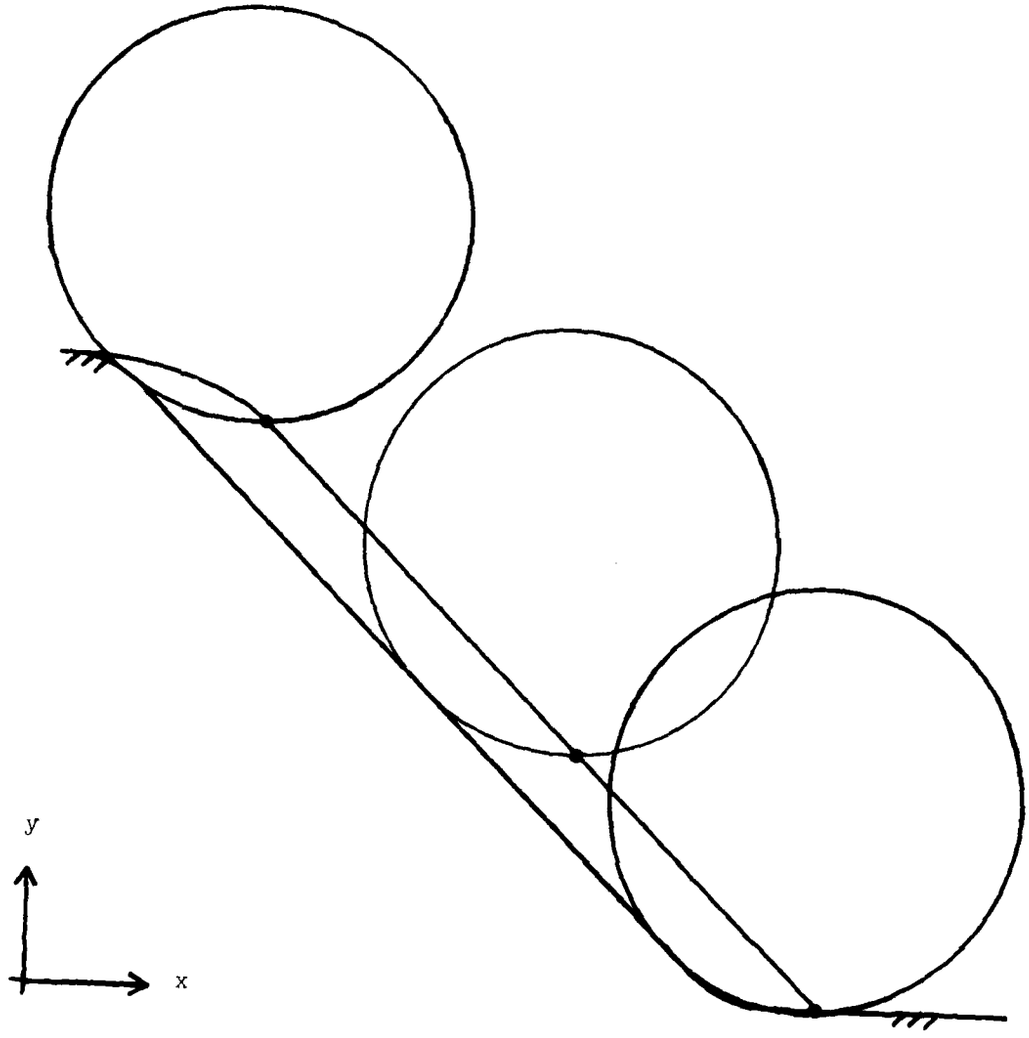


Figure 6.10. Wheel trajectory defined by lowest points on wheel with no terrain penetration.

wheel-terrain contact and wheel center. Thus, there will be a component of force tending to impede leftward motion of the wheel and, if not otherwise supported, the wheel will move down the slope. The angle that this force makes with respect to the positive y axis is shown in Figure 6.11. The curves of Figures 6.9 and 6.11 are provided to the program in the form of cubic spline functions.

The normal and frictional components of force acting on the wheel remain to be determined. As noted earlier, if global coordinates x_i and y_i locate the center of a roadwheel and if $g(x_i)$ is the trajectory height beneath the wheel, as in Figure 6.9, then wheel penetration may be approximated by

$$p_i = [g(x_i) + r_w - y_i] \tag{6.1}$$

where r_w is the roadwheel radius. Normal force angle, from the function h(x) of Figure 6.11, is

$$\alpha_i = h(x_i) \tag{6.2}$$

Figure 6.7 illustrates a typical nonlinear force-displacement relation between a roadwheel and ground. Let this curve be represented by the function $f(x_i)$, so the total vertical force acting on the wheel, as shown in Figure 6.12, becomes

$$F_s = F_{ny} + F_{fy} = f(p) \tag{6.3}$$

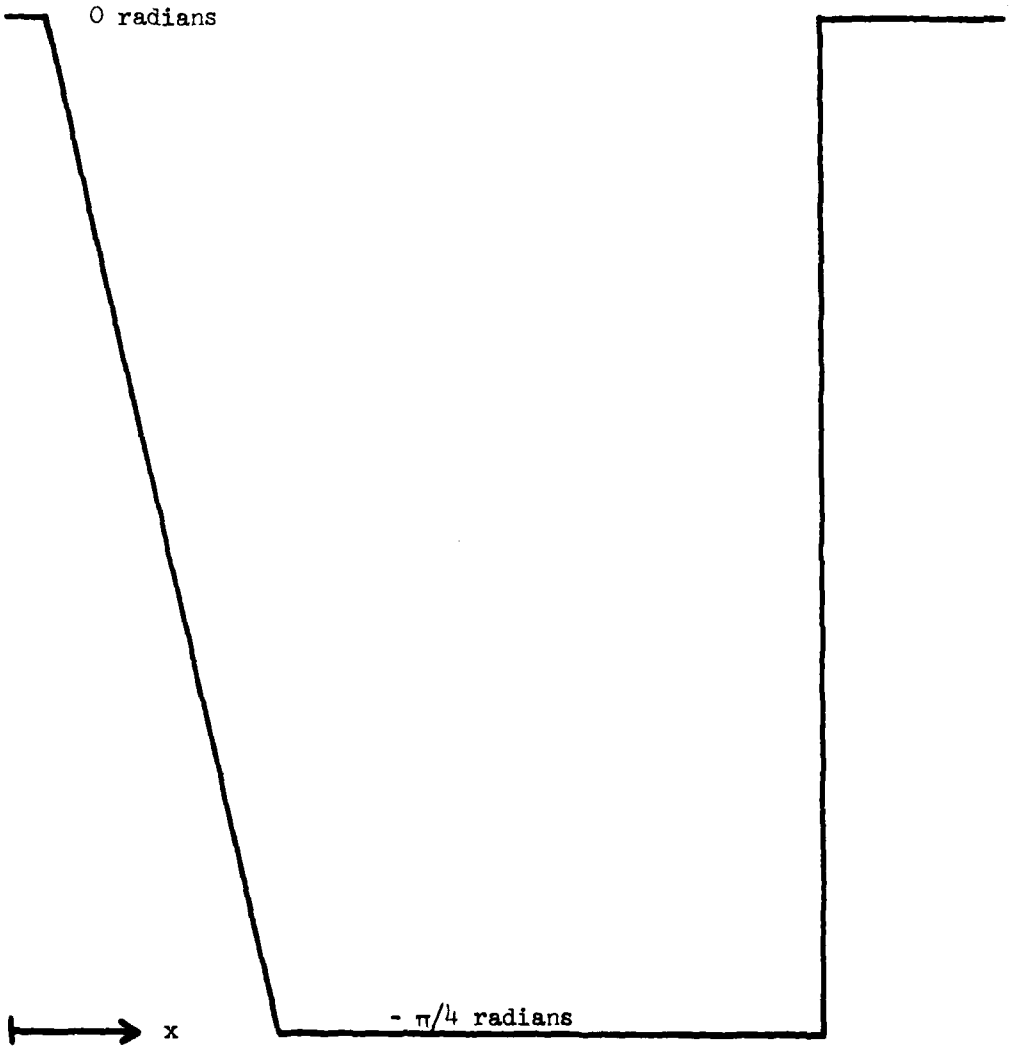


Figure 6.11. Direction of a line passing through point of wheel-terrain contact and wheel center.

where F_{ny} and F_{fy} are the vertical components of the normal, $F_n \geq 0$, and frictional F_f forces, respectively. The subscript i has been dropped for convenience. In terms of the angle α and the sign conventions of Figure 6.12, the following expressions hold:

$$\begin{aligned} F_{nx} &= - F_n \sin \alpha \\ F_{ny} &= F_n \cos \alpha \end{aligned} \tag{6.4}$$

$$\begin{aligned} F_{fx} &= F_f \cos \alpha \ \text{sgn (slip)} \\ F_{fy} &= F_f \sin \alpha \ \text{sgn (slip)} \end{aligned} \tag{6.5}$$

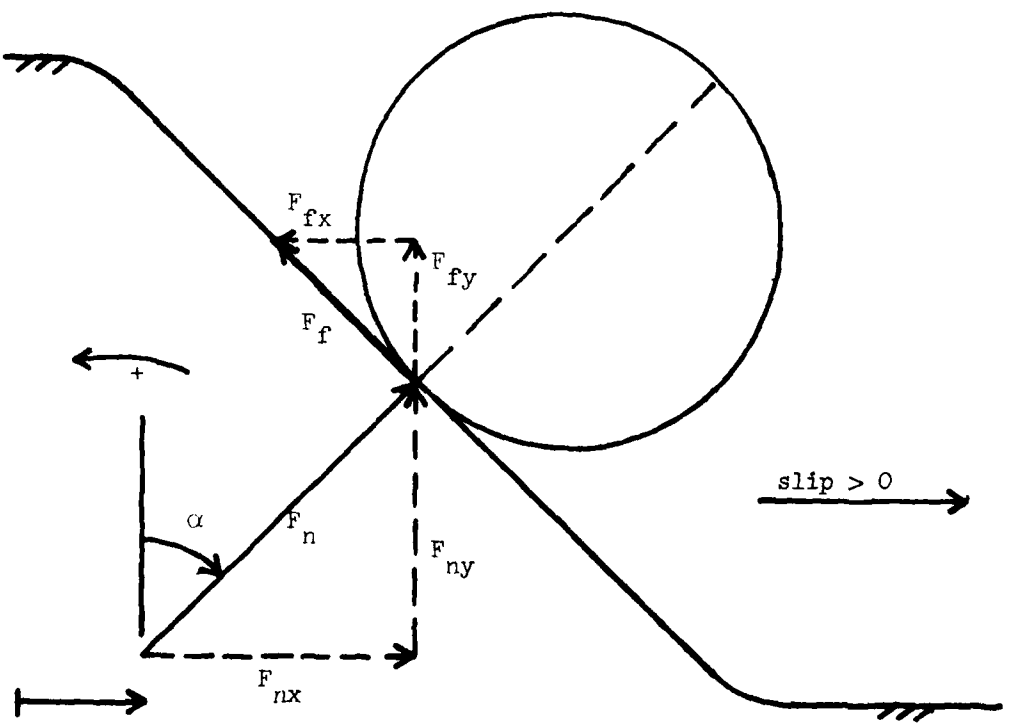


Figure 6.12. Terrain-wheel contact forces.

where positive slip occurs if the wheel slides to the right. The wheel may not slide, in which case there is some force F_f' such that

$$|F_f'| < |F_f| \tag{6.6}$$

If the coefficient of sliding friction is μ, then the resulting frictional force is

$$F_f = \min\ (|F_f'| \text{ or } \mu F_n)\ \text{sgn (slip)} \tag{6.7}$$

Equations 6.3 to 6.7 can be combined to derive an expression for the frictional force, in terms of the vertical force F_s,

$$F_f = \min\ [|F_f'| \text{ or } (\mu F_s)/\{\cos\alpha - \mu \sin\alpha\ \text{sgn (slip)}\}]$$

$$\times\ \text{sgn (slip)} \tag{6.8}$$

The vehicles propel themselves along the terrain, so driving forces are applied to the track and roadwheels to achieve this goal. For simplicity, it is assumed that the vehicle is to travel at a constant velocity across the terrain. Denote this velocity as v_r and the desired lateral displacement as $S_{rj} = S_{0j} + v_r\ t$, $j = 11,12$, where S_{0j} is the global location of a reference point on the j^{th} vehicle at time, $t = 0$. Locating this point at ground level and at the center of the vehicle will secure an approximately constant

ground speed. Let v_j be the actual velocity of this point on body j at any instant t, then the actual displacement of this point is

$$S_j = S_{0j} + \int_0^t v_j dt \tag{6.9}$$

Track tension is developed according to error in the desired vehicle position and velocity as

$$T_j = K_d(S_j - S_{rj}) + K_v(v_j - v_r) \tag{6.10}$$

where the constants K_d and K_r are chosen to characterize the vehicle power train. Observe that T_j may be positive (vehicle lags behind desired position) or negative (vehicle is ahead of desired position). If $T_j > 0$, T_j is applied to track elements S16 and S22. If $T_j < 0$, $-T_j$ is applied to track elements S11 and S17. The maximum tractive effort is limited to the maximum force developed by a slipping track.

The sequence of steps to evaluate track driving forces for the front vehicle are described. If $T_j > 0$, set $F' = T_j$. Find F_s for sprocket C, as shown in Figure 6.4, using Equation 6.3. Assume slip > 0 (Figure 6.12), since the vehicle is being propelled to the left. If $F_s = 0$, the sprocket is not in contact with the terrain. Therefore, no frictional tractive force is developed beneath the sprocket, so the entire force F' is transferred to track element S16.

If sprocket C is in contact with ground, $F_s > 0$. Therefore, determine the frictional tractive force F_f acting between sprocket

and ground, using Equation 6.8. Apply F_f to the sprocket at the point of sprocket-terrain contact, directed along the tangent to the terrain surface at this point. The track driving force F' is now reduced by F_f and is applied to track element S16. The process is then repeated in a similar manner for each successive roadwheel, rear to front. Calculate F_s for roadwheel (B5) from Equation 6.3 and F_f from Equation 6.8. Apply F_f to roadwheel (B5) at the point of roadwheel-terrain contact, directed along the tangent to the terrain surface. Again reduce F' by F_f, and apply this force to element S15. Continue this process until F' has been reduced to zero, or until sprocket D has been passed. If F' becomes zero, the vehicle is not traction limited. If $F' > 0$, the track is slipping and the desired reference position, S_{rj} in Equation 6.10, cannot be attained; so set $S_{O_j} = S_j - v_r t$ and $S_{rj} = S_j$, the attained vehicle displacement.

If $T_j < 0$ in Equation 6.10, the process is reversed. That is, drive forces are applied from front to rear. First check sprocket D, roadwheel (B1), etc. for contact and apply appropriate forces. The vehicle is being driven to the right in this case and slip is assumed to the left (Figure 6.12), hence it is negative.

The significant feature of this model is that track slippage at each wheel is allowed if there is enough force differential in the track to overcome local friction μF_s. The remaining road wheels, once F' becomes zero, are assumed to have zero frictional forces acting. This will not be true for the actual system, especially on

rugged terrain. However, these forces will be directed randomly, some forward and some rearward, due primarily to terrain irregularity, and will not contribute significantly to vehicle dynamics.

(d) Elements of the Articulation Control Model

The hydraulic actuator is modeled by using the actuator of element S23. Actuator force ($F_{O_{23}}$), displacement ℓ_{23}, and velocity v_{23} are standard state variables. These variables are coupled to the control system equations to complete the model.

Figure 6.13 contains the block diagram representation of the electro-hydraulic control system [47, 48] and depicts the inter-relationship between the servo control system and the dynamic mechanical model, where:

$F_R(s)$ = operator supplied fore and aft force to the control stick, lb_f

d_1 = length of control stick, inches

$\theta_R(s)$ = angular displacement of control stick, radians, $|\theta_R| < .436$ radians

$T_L(s)$ = control stick limiting torque, in-lb_f

τ_c = control system time constant, sec

K_{RP} = potentiometer gain, volts/radian

$R(s)$ = reference voltage signal from control stick potentiometer, volts

K_1 = amplifier gain, volts/volt

$z(s)$ = actuator displacement, inches

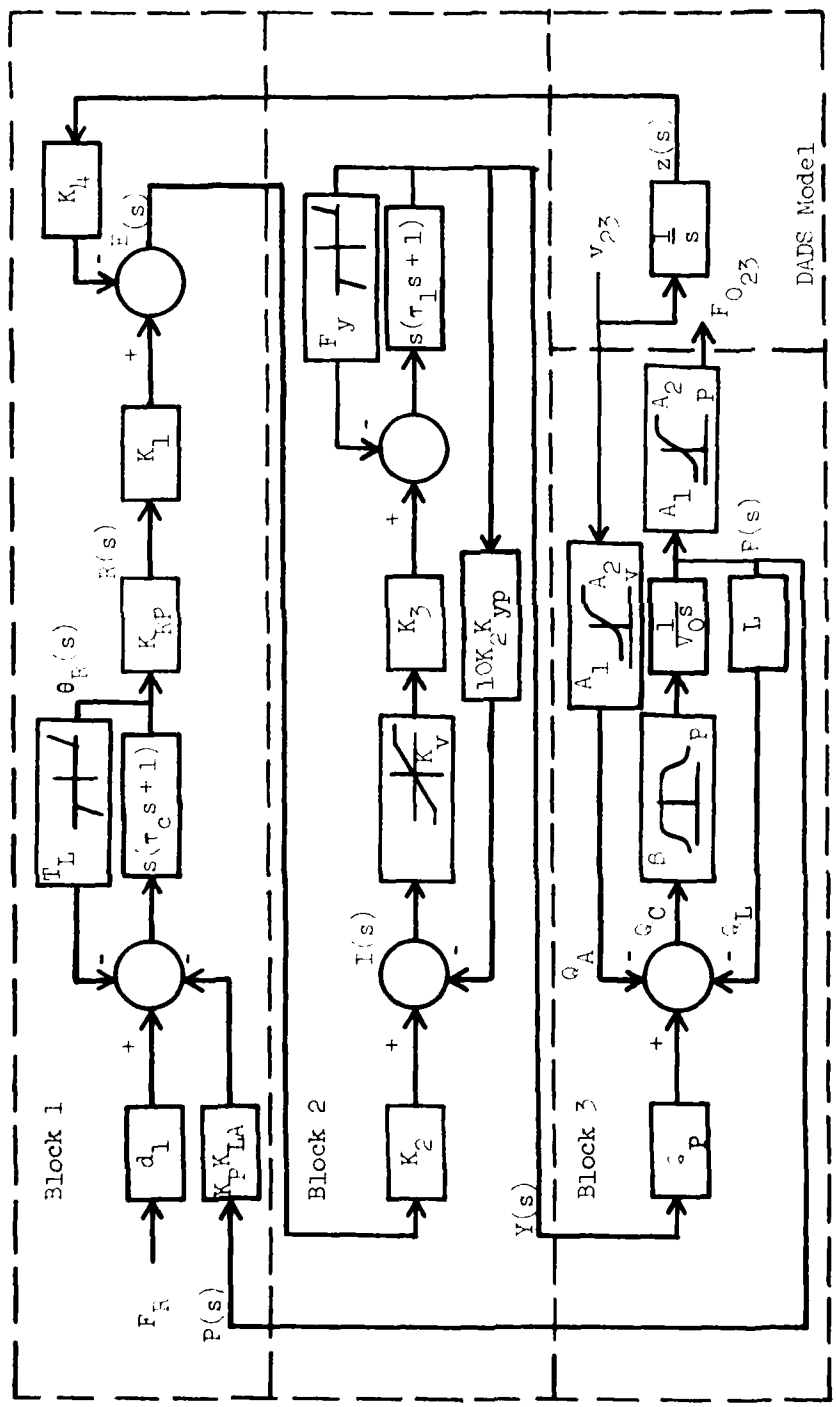


Figure 6.13. Control system diagram.

$\ell_{23}(s)$ = total actuator length, inches

$\ell_{0_{23}}$ = reference actuator length, inches

$v_{23}(s)$ = actuator velocity, in/sec

K_4 = actuator potentiometer feedback gain, volts/in

$E(s)$ = error signal voltage, volts

K_2 = amplifier gain ma/volt

$I(s)$ = amplifier current, ma

K_v = servo valve flow gradient, GPM/ma

K_3 = pump stroke mechanism velocity gradient, % stroke/sec

τ_1 = servo valve time constant, sec

$Y(s)$ = pump yoke displacement, % stroke

$F_y(s)$ = pump yoke limiting force, lb_f/in

Q_p = pump flow gradient, $(in^3/sec)/(\%\ stroke)$

Q_T = pump flow, in^3/sec

Q_A = load flow, in^3/sec

Q_L = leakage flow, in^3/sec

Q_C = flow due to compressibility, in^3/sec

A_1 = actuator area - rod end, in^2

A_2 = actuator area - cylinder end, in^2

β_e = effective bulk modulus including system compliance, psi

V_0 = entrained volume of hydraulic fluid

L = combined system leakage coefficient, $(in^3/sec)/psi$

$P(s)$ = system pressure at the actuator, psi

$F_{O_{23}}$ = force developed in the actuator, lb_f

K_p = pressure transducer gain, mv/psi

K_{LA} = control stick force feedback gain, in-lb_f/mv.

The articulated vehicle control system may be divided into three major blocks. As shown in Figure 6.13, block 1 generates a reference voltage that ultimately leads to a given hydraulic actuator displacement. The major component in this block is the mechanical hardware that interfaces with the operator. Block two contains the hydraulic pump yoke and servo valve control system. In response to an error signal from block 1, this system controls direction and fluid flowrate in the hydraulic pump. Block 3 characterizes dynamic response of the hydraulic system, including the actuator connected to the articulated vehicle model.

The control stick, shown in Figure 6.14 [47], rotates about a fixed point. A direct current linear actuator, shown in its neutral position, is coupled to an extension of the control stick d_2 by springs K_1 and K_2. These springs provide restoring moments to maintain the control stick in its neutral position. In response to pressure buildup in the hydraulic actuator, a voltage is developed across the linear actuator, causing springs K_1 and K_2 to shift either left or right. An increased moment is then developed on the control stick to oppose the operator supplied force, thus providing force feedback to the operator. In addition, mechanical stops prevent more than $\pm$ 25° rotation of the control stick. The dynamics of this system, including inertial effects, is included in block 1. Control

stick response is determined by three moments: operator's moment, $(F_R \cdot d_1)$; pressure feedback moment, $(-K_p K_{LA} P)$; and mechanical stop limiting torque, $[- T_L(\theta_R)]$.

A rotary potentiometer connected to the control stick develops a reference voltage that is proportional to angular rotation θ_R. This signal is then amplified and combined with a hydraulic actuator displacement signal $(- K_4 z)$ to obtain an error signal E. This error signal is then amplified and passed to the pump yoke control system.

Servo valve and pump yoke characteristics are illustrated in block 2 of Figure 6.13. Significant nonlinear effects in the system

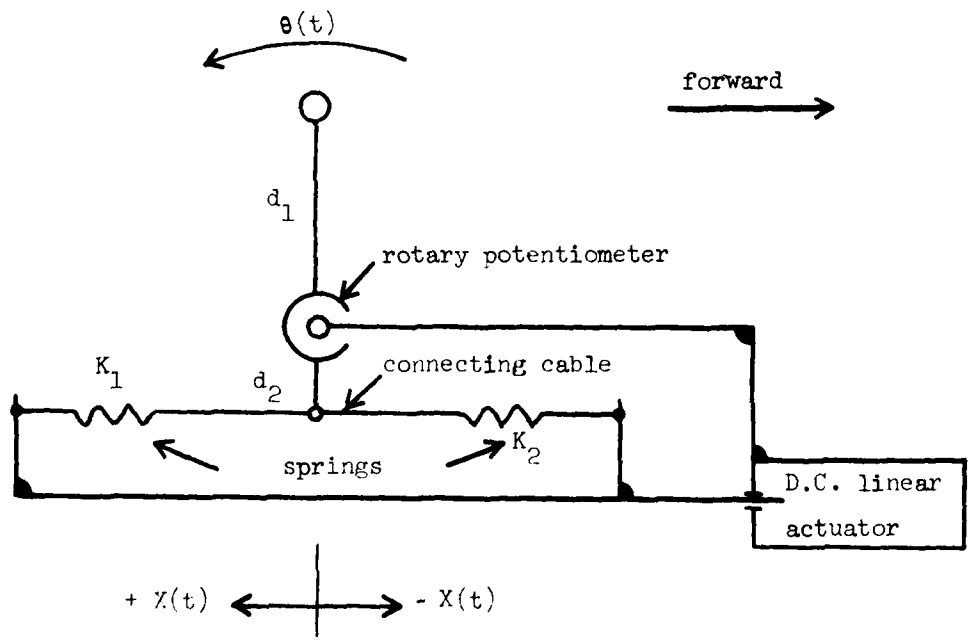


Figure 6.14. Control stick and force actuator.

are saturation of the servo valve flow control K_v and mechanical limitations on pump yoke displacement F_y. Inertial effects in this block introduce additional delay in overall system response.

Block three outlines the hydraulic flow model. Flow rates within the system are moderate, so pressure drop within the lines is small. This simplifies the model significantly. The pump supplies fluid volume flow rate Q_p, to make up for leakage losses Q_L, compressibility Q_C, and actuator volume displacement rate Q_A.

The system contains a significant amount of flexible tubing, which results in reduced system compliance, hence a small effective bulk modulus. Pressure limiting is obtained by further reduction of the effective bulk modulus, as relief valve blowoff begins. Actuator flow rate from the pump is determined by the product of actuator cross-sectional area A and velocity v. The area A_1 on the rod end of the actuator is smaller than the piston end area A_2. Thus, area is adjusted appropriately as a function of actuator velocity to achieve the proper flow rate. Force developed in the actuator also depends on which side of the piston pressure is developed. Thus, area is adjusted as a function of pressure, for the purpose of calculating actuator force. The controller consists of two identical hydraulic systems, each driving an actuator. In this simulation, only one hydraulic system is modeled, so the effective actuator force is doubled to account for two actuators.

(e) Crossing a Three-Foot Obstacle - Numerical Results

The three-foot step obstacle shown in Figure 6.8 is selected to test the program's ability to adequately describe system dynamics. In addition, the effect of force and position feedback on vehicle performance, while negotiating a significant obstacle, is to be investigated. A velocity of 40 inches/second (2.27 miles/hour) is selected as the obstacle crossing speed. This is close to the design specifications of minimum 2.5 miles/hour over a 2.5 foot obstacle.

Contrary to usual practice, the vehicle is not put into an initial pitch up attitude before encountering the bank, but is forced to propel itself up the side of the bank. The vehicle is run in a hands off mode, because it is difficult to program operator response to the complex dynamic environment.

Three simulations of obstacle crossing are performed. The first case is with force and position feedback active, the second is with force feedback deactivated, and the third is with the entire control system deactivated. In the last situation the control system is passive, except that the hydraulic pump is providing enough fluid flow rate to replace leakage losses. A sequence of computer generated line drawings of vehicle position at one-second intervals for the three cases is shown in Figures 6.15-6.17.

The following observations can be made from these figures. Force and position feedback in Figure 6.15 tend to minimize actuator

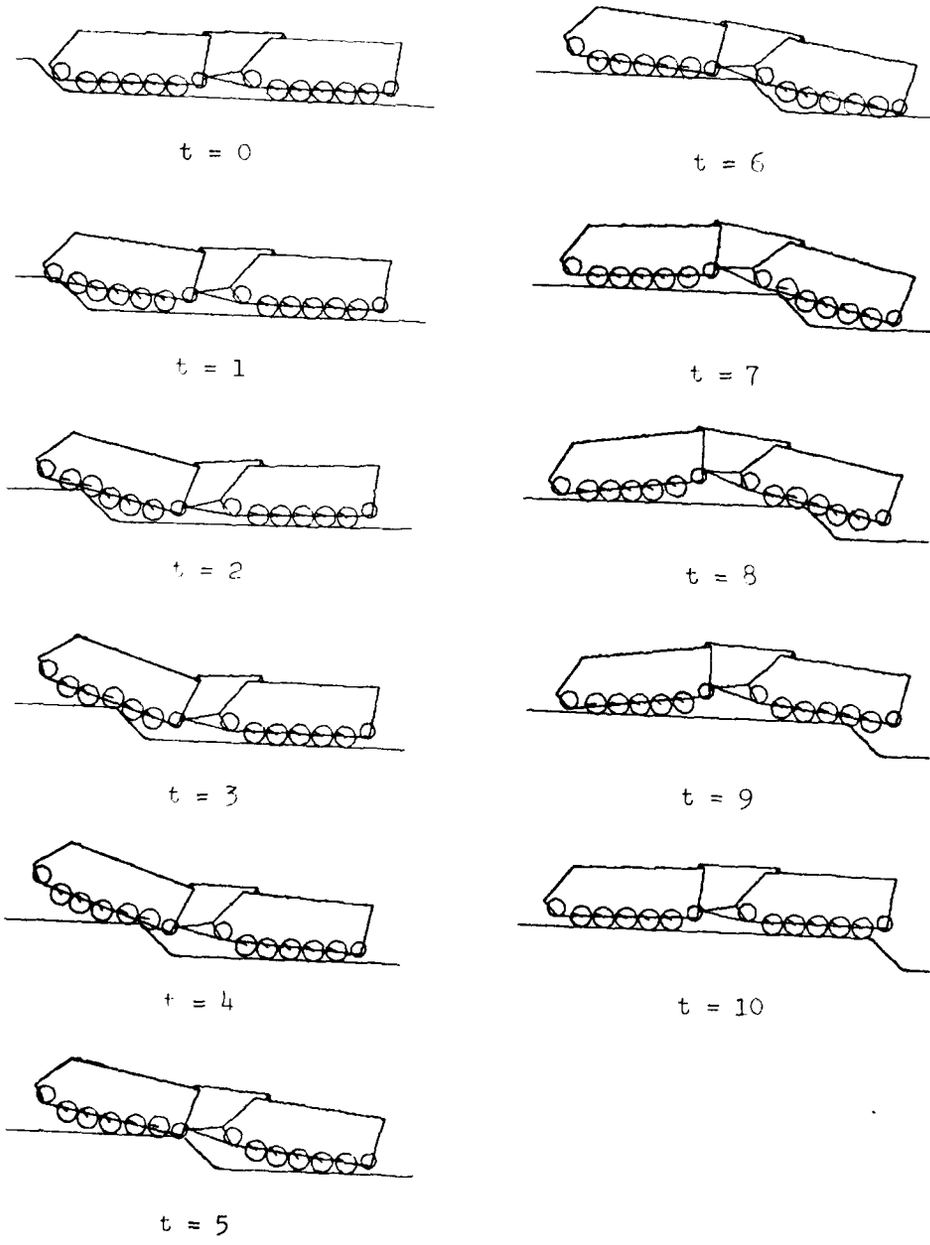


Figure 6.15. Obstacle crossing with force and position feedback active - ten second simulation at 40 inches/second.

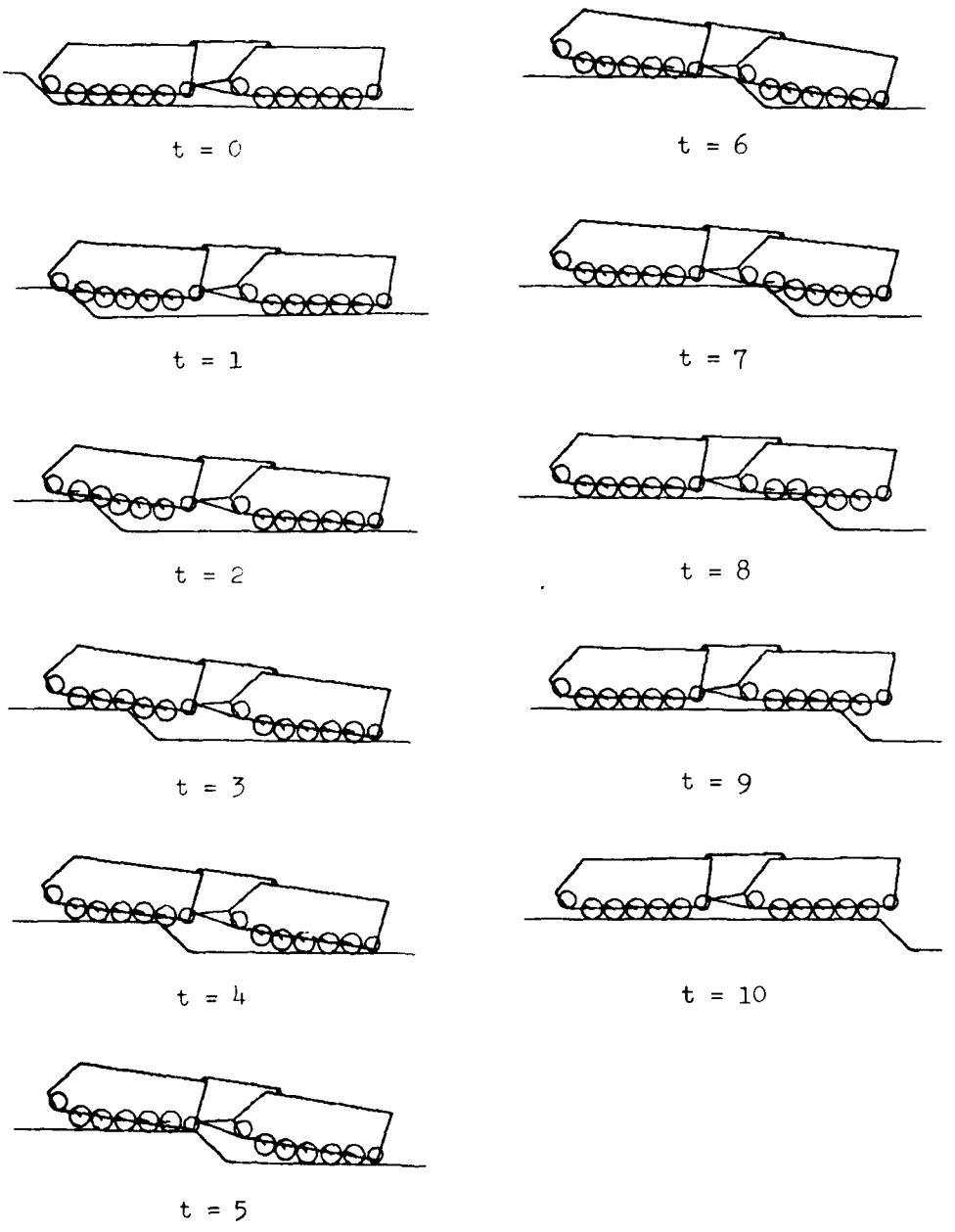


Figure 6.16. Obstacle crossing with force feedback deactivated - ten second simulation at 40 inches/second.

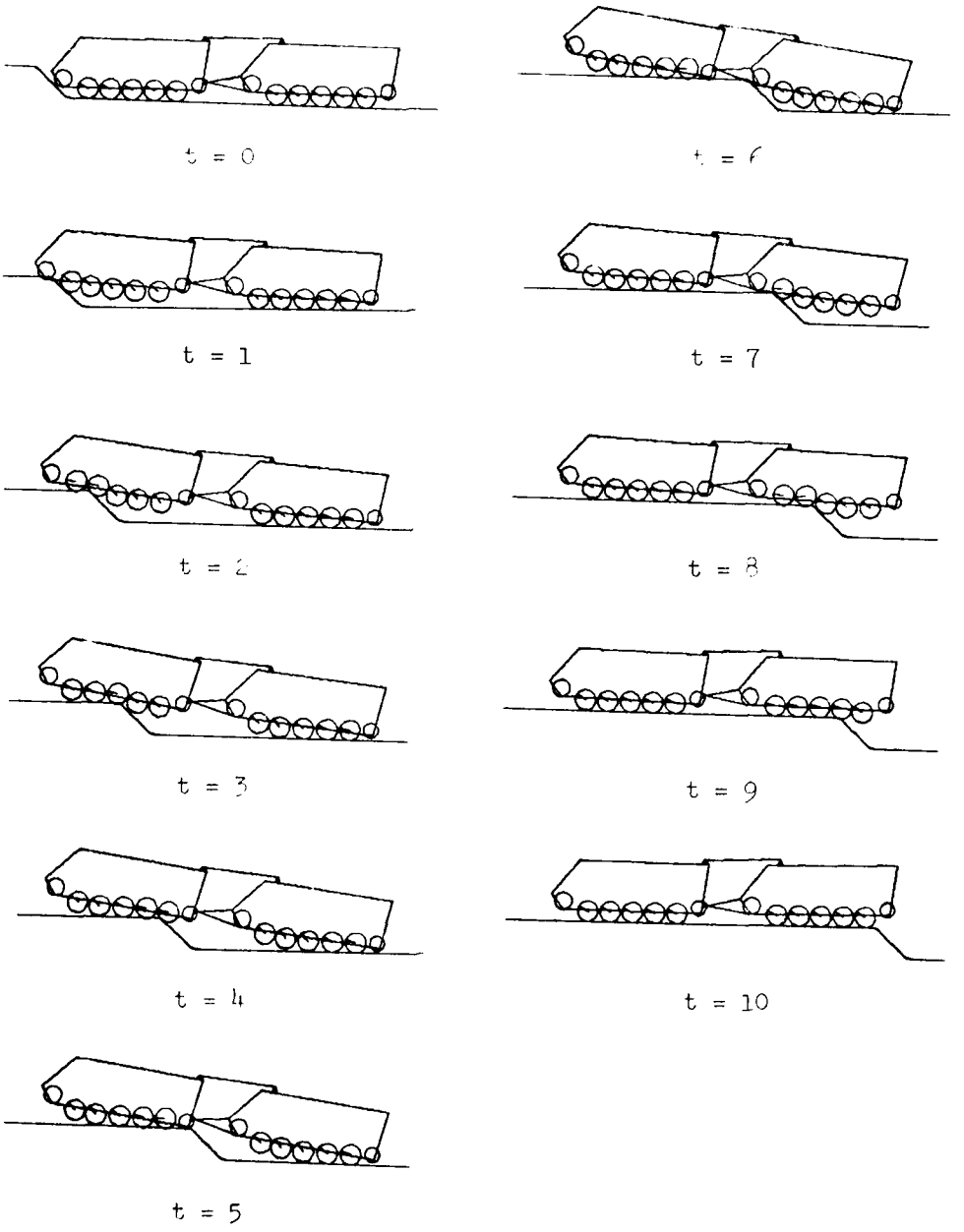


Figure 6.17. Obstacle crossing with the control system deactivated - ten second simulation at 40 inches/second.

stiffening of the coupled vehicle system. This will be the most comfortable and stable situation since the vehicles tend to conform to the terrain.

Position feedback alone in Figure 6.16 tends to maximize vehicle rigidity in the hands off mode because the control system continually works to maintain minimum departure from the zero relative pitch position. With operator assistance, however, one might expect vehicle response approaching that of case 1.

The passiv model in Figure 6.17 is slightly more flexible than case 2, because of system compliance. An ideal passive system for large obstacle crossing would be one that allows free or partially restricted fluid flow in the actuator, through a simple bypass.

To understand more clearly how force feedback works to minimize actuator pressure, it is shown plotted versus time in Figure 6.18. Note that positive pressure implies that pressure acts on the piston end of the cylinder and is zero on the rod end. Negative pressure implies that pressure acts on the rod end of the cylinder and is zero on the piston end. This is a convenient way to mathematically represent system pressures. Pressure is generally positive for the first 3 to 4 seconds (pressure on piston end), as the front vehicle is forced up and over the step. It then remains negative between 4 and 10 seconds for case 2, since the front vehicle is now being supported by the actuator. Pressure is essentially negative between 4 and 8 seconds and positive between 8 and 10

seconds, when force feedback is active. It is clear that force feedback substantially reduces actuator pressure, as is expected.

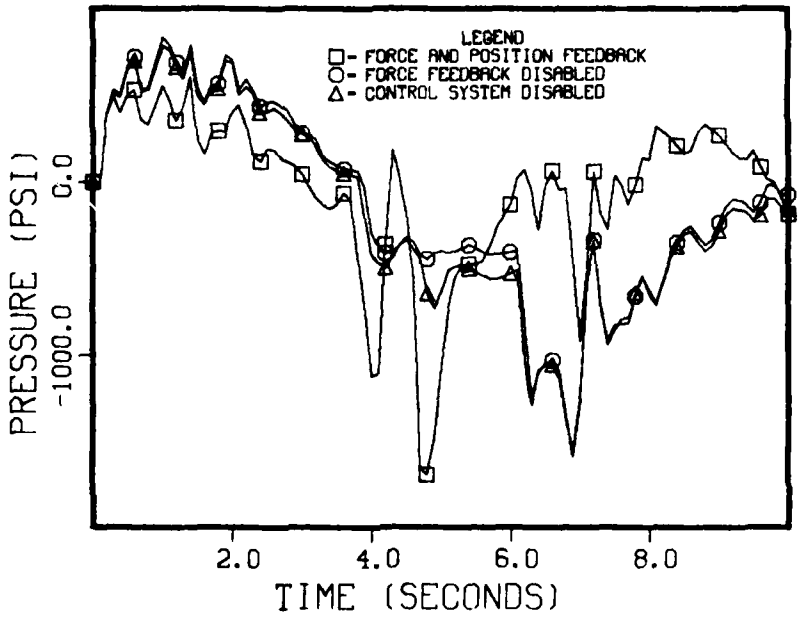


Figure 6.18. System hydraulic actuator pressure.

The reason for reduced pressure is made clear by looking at Figure 6.19, angular displacement of the control stick, and Figure 6.20, hydraulic pump yoke displacement. When actuator pressure initially goes positive, the control stick is driven into the full pitch up position (- 25°). The control-stick signal then induces a large pump yoke displacement, causing hydraulic fluid flow from the high pressure end. When pressure switches to the opposite side of the piston, control stick and pump yoke displacements quickly switch to their opposite extremes, again to compensate for large actuator pressure.

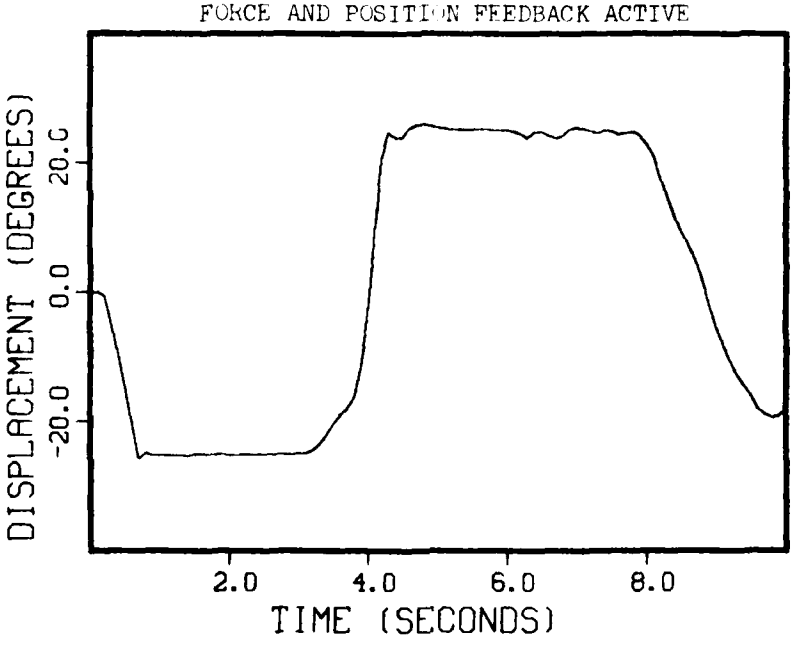


Figure 6.19. Angular displacement of the control stick (negative = pitch-up).

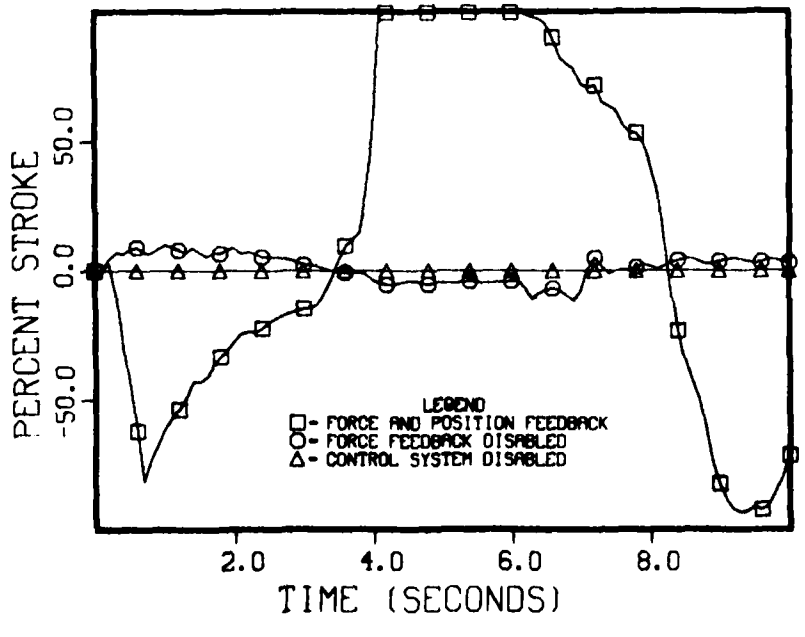


Figure 6.20. Hydraulic pump yoke displacement (percent full stroke).

(f) Model Verification and Discussion

The articulated vehicular system simulated in this study was assembled and tested by the Army in the early to mid 1970's [47, 48]. A large amount of field and design test data was accumulated. Upon termination of the project, the vehicles were disassembled and returned to regular service. Several years later the DADS computer program was developed and it was recognized that this vehicle would make an excellent test model to demonstrate DADS analysis and design potential. Unfortunately, some important system parameters were not measured or were not documented. Therefore, it was necessary to estimate some parameters in order to complete the model.

Several tests that were documented [47, 48] measured transient pitch response to various pitch-up and pitch-down command signals. The test vehicles were stationary and on a level surface. Simulated vehicle pitch rates and hydraulic pressures under identical conditions compared favorably with field test data [53]. Discrepancies, however, did provide suggestions for improvement in the initial model, i.e., the need for increased hydraulic system compliance, track supportive effects, and frictional effects. It is believed that these initial simulations, for which field test data was available, have resulted in a more representative mathematical model.

Unfortunately, such extensive test data has not been recorded for obstacle crossing experiments. However, 16 mm filmed records were made of various obstacle crossing tests. One such test involved

a three foot step obstacle where the vehicle was placed in an initial pitch-up attitude allowing the front end to negotiate the obstacle. The vehicle then propelled itself onto the step with some operator assist. Test vehicle velocity was comparable to the velocity in this simulation.

A 16 mm computer generated, animated, film of the dynamic simulation (composed of images similar to those in Figures 6.15-6.17) was compared to the filmed test results mentioned above. The agreement was significant. Dynamic response of the vehicles as the roadwheels encountered the step was very similar. The time lag in the vehicle's ability to conform to terrain was apparent in both cases, but not as significant in the field test results. This could be due to operator assist or to less inertia in the actual control system than was modeled, allowing faster system response.

This example is presented in detail to emphasize the many important considerations one must make in order to have even a simple representative model of a complex system. It is shown that most of the nonstandard effects are conveniently introduced by modifying existing standard program elements and employing standard state variables. Thus, the magnitude of equation development and programming effort has been kept to a bare minimum. Even more significant than the efficiency gained by the analysis methods developed herein is the fact that derivatives of generalized forces are not required. The program described in Chapter 2 requires

derivatives of generalized forces, which substantially increases the amount of program preparation and debugging effort.

The independent generalized coordinates identified by the program for the above simulations are x_1, x_5, y_1 to y_{12}, and φ_1 to φ_{10}. One could make intuitive arguments based on mechanical or mathematical principles as to why certain variables are selected over others. However, since the selection process depends so strongly on constraints, one must thoroughly understand each constraint type and how it interacts with others, if an acceptable set is to be selected. This emphasizes the importance of automatic variable selection.

It is emphasized that various vehicle and control system parameters are estimated, because measured values are not available. Thus simulated response will vary somewhat from actual system response. The primary purpose of this research is to demonstrate that DADS can be effectively employed to obtain dynamic response of complex mechanical systems. Therefore, it is demonstrated to be an invaluable analysis and design tool.

6.3 A Mechanism With Intermittent Motion

(a) Introduction

A precision weapon mechanism was selected for simulation to illustrate use of the pieced interval analysis methods developed in Chapter 5. Previously, methods of intermittent motion analysis have used pieced interval analysis, in which the analyst writes the

equations of motion between times at which discontinuous events occur [43]. Momentum balance equations were also written to account for velocity discontinuities that occur in various system configurations. Numerical integration was halted at points of discontinuity, new initial conditions were formulated, and integration was then continued. A limitation of this method of analysis is the level of effort required to write system equations that are valid in intervals between events whose ordering is not generally known before the analysis is begun. Thus, the analyst is required to write equations and computer code for all ordering of logical events that may conceivably occur.

One method that has been used to alleviate the foregoing difficulty is to use Heaviside step functions that define logic associated with the events occurring during intermittent motion. These discontinuous functions may then be smoothed to provide a set of governing differential equations of motion [54]. This procedure can be justified on the basis of distribution theory [55, 56] and has been successfully employed in weapon mechanism dynamics [57]. The distribution theoretic method has been used in conjunction with the DADS computer program to automatically generate the system equations of motion [58], by defining "logical spring-dampers" that account for certain aspects of intermittent motion [21]. These analyses are plagued by integration failure and inefficiency, while integrating through abrupt transition regions. This deficiency has now been eliminated by employing the logical

events predictor developed in Chapter 5 to locate such events before they are encountered. Thus, the integration algorithm can be prepared to handle the events or alternate integration methods can be employed.

Alternately, one may choose to replace (or supplement) the smoothed continuous events by discontinuous events, in which momentum balance is employed. Costly numerical integration through transition regions is then replaced by efficient algebraic solution of momentum balance equations. Following momentum balance, one may elect to retain the continuous elements, if desired, to represent interference between adjacent bodies.

(b) Description of an Automatic Cannon System

An automatic weapon mechanism shown in Figure 6.21 consists of four main masses; the receiver R (assumed rigidly attached to the inertial reference frame), the barrel assembly B, the sleeve S, and the sear SR. The sequence of system operation for a cycle is as follows:

The barrel assembly is initially moving to the right at 40 inches per second at the instant shown in Figure 6.21. A round is inserted into the shaded region of the chamber as the barrel is moving forward. A link pq is connected at point q to a sleeve that slides along the outer surface of the barrel and to a pin p located at the intersection of the two cam slots to control sleeve position on the barrel. The receiver-cam is attached to the receiver and the

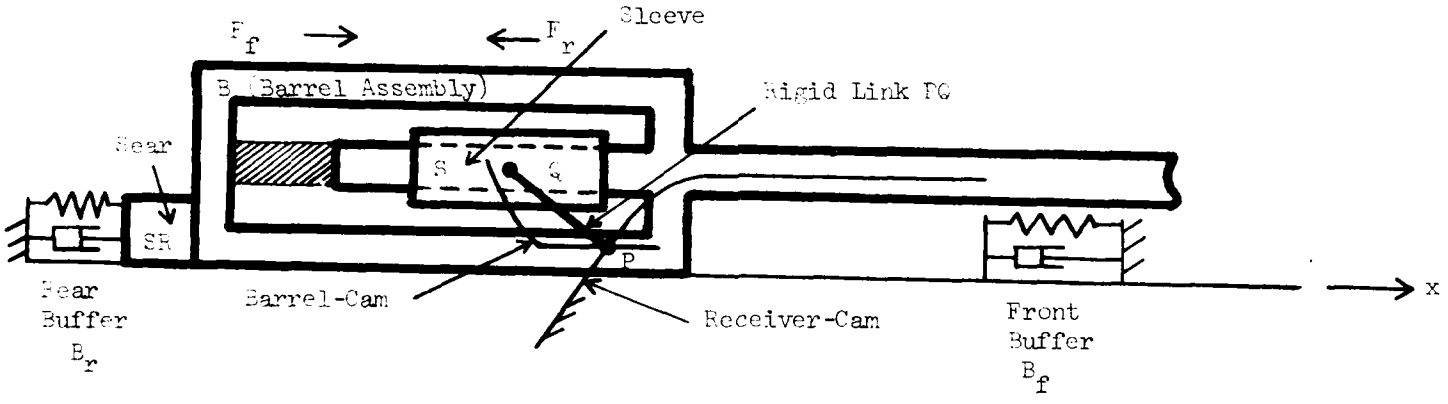


Figure 6.21. Cannon system shown in positions where barrel leaves or impacts sear.

barrel-cam is attached to the barrel. As the barrel moves, the point of cam slot intersection changes, forcing the pin to move.

Figure 6.22 shows the configuration when the pin just begins to move upward as the barrel is moving to the right. This upward motion drives the sleeve to the left, relative to the barrel, hence encasing the round. Eventually (Figure 6.23), the pin reaches the highest point on the receiver cam, and the sleeve in its rearmost position relative to the barrel, securely locks the sleeve in position to form the chamber of the weapon. The sleeve now moves with the barrel.

Next, as shown in Figure 6.24, the barrel contacts a front buffer. This buffer is designed to bring the barrel safely to rest in the event of a round misfire. Shortly thereafter (Figure 6.25) the round is fired, developing an impulse on the barrel that reverses its direction of travel.

The sequence of operations is then reversed (Figures 6.24, 6.23, and 6.22). After the sleeve has been retracted, the spent round is ejected and the barrel impacts the sear (Figure 6.21). The sear and barrel, moving rearward together, are then brough to rest by a rear buffer (Figure 6.26). The sear and barrel are then driven forward by the buffer spring until the sear impacts a stop on the receiver (Figure 6.21). The barrel separates from the sear and a cycle is complete (ready for the next round in the automatic fire mode), or the barrel remains attached to the sear to terminate execution.

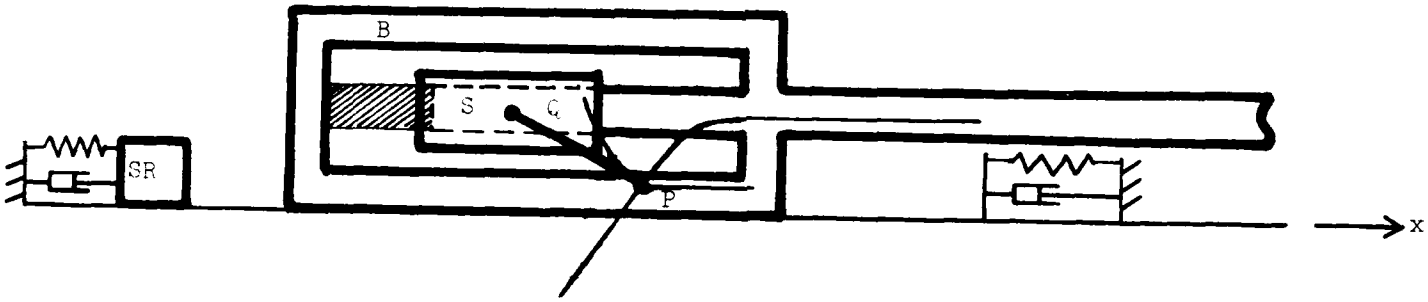


Figure 6.22. Position where sleeve starts to move rearward with respect to barrel or comes to rest with respect to receiver.

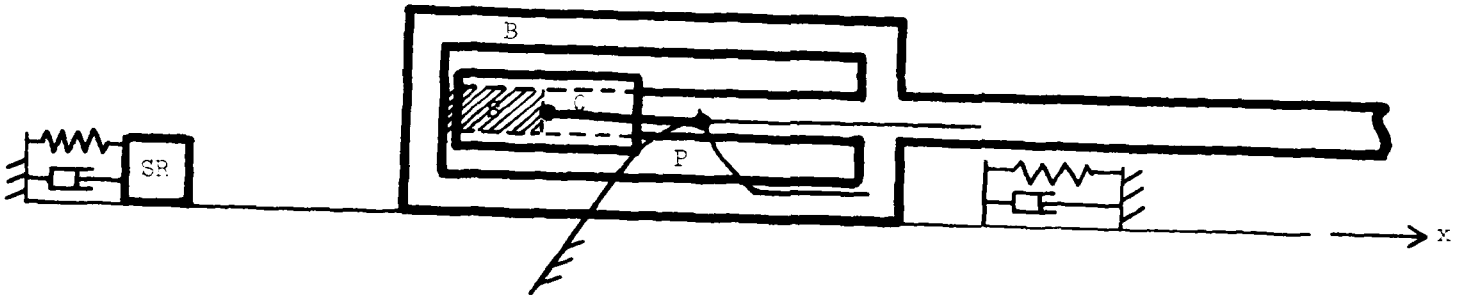


Figure 6.23. Position where rearward motion of the sleeve with respect to barrel ceases or begins.

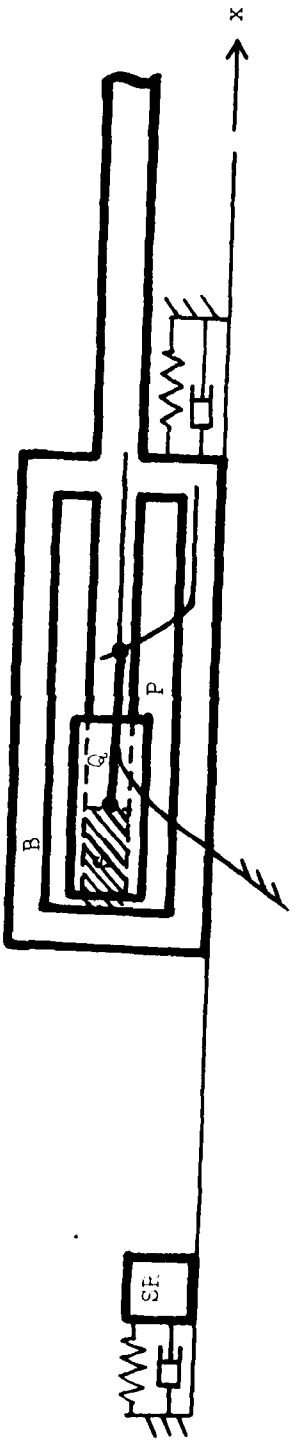


Figure 6.24. Position where barrel contacts or leaves front Buffer.

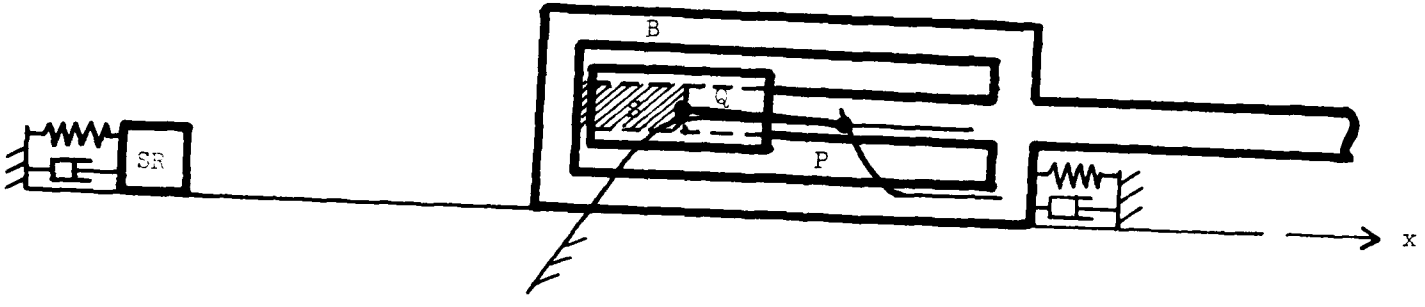


Figure 6.25. Firing position.

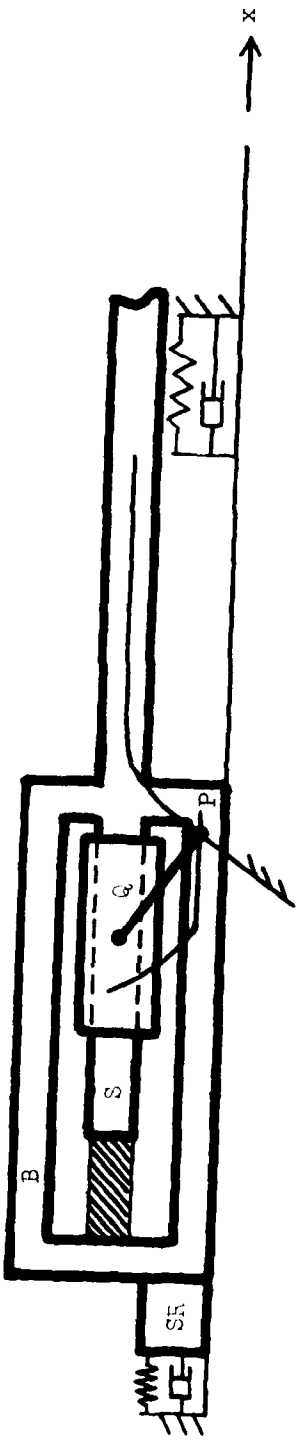


Figure 6.26. Barrel assembly at rearmost position.

(c) Elements of the Model

The weapon mechanism consists of five rigid bodies: receiver, barrel, sear, sleeve, and pin, numbered in this order. The sear and barrel each move on the receiver, along translational joints, and the sleeve moves on the barrel along a translational joint. Mass of the barrel, sleeve, and sear are taken as 2.919, .2368, and .2044 slug-feet/inch, respectively. Other masses and rotational inertias are ignored or are not important, since no rotation is allowed. The sleeve is connected to the pin by a massless link pq. For the barrel position shown in Figure 6.21, the link is initially at a 45° angle and it moves to a horizontal position in Figure 6.23, as the pin rides up the cam slots.

The pin-cam constraints are represented by cubic spline curve fits to digitized data taken from the mechanism. The vertical position of the pin relative to the receiver is written as a function of the horizontal position of the pin relative to the receiver as

$$y_P - y_R - f_R(x_P - x_R) = 0 \tag{6.11}$$

where f_R is the functional spline representation of the cam path on the receiver. Likewise, a cam path on the barrel is written as

$$y_P - y_B - f_B(x_P - x_B) = 0 \tag{6.12}$$

These equations and their first and second time derivatives are appended to the standard constraint formulation, to complete the model.

Two forces, F_f and F_b, drive the barrel during its forward (counter recoil) and rearward (recoil) motion, respectively. A front buffer B_f and a rear buffer B_r slow the barrel assembly during extreme displacement. Both front and rear buffers are designed to produce constant retarding forces.

(d) Logical Events

Logical times t_i at which impact or other irregularities of intermittent motion occur are introduced as an integral element of the dynamic model. Between these times, the motion and acceleration of the system is continuous. At these times, discontinuities in velocities and acceleration, changes in system constraints, and mass capture or release can occur. These logical times are functions of the system state and are determined as the simulation progresses. Logical times will now be defined for the firing from run-out mode of weapon operation:

(1) $t_0 = 0$ (Figure 6.21): The barrel assembly B, in automatic fire, passes the sear position with velocity $\dot{x}_2 =$ 40 in/sec. Initial starting point is not considered as a logical event. A forward driving force $F_f = 1600\ lb_f$ acts on the barrel.

(2) t_1 (Figure 6.24): The barrel B contacts the front butter and B_f = - 6900 lb_f becomes active. Restart integration because of discontinuous acceleration.

(3) t_2 (Figure 6.25): The charge is ignited. An impulse of - 880 lb_f-sec acts on the barrel B. Perform momentum balance to obtain new velocities. F_f is deactivated and drive force F_b = 2000 lb_f is activated. Restart integration.

(4) t_3 (Figure 6.24): The barrel B breaks contact with the front buffer and B_f = 0 lb_f. Restart integration.

(5) t_4 (Figure 6.21): The barrel B impacts and captures the sear SR which was locked to the receiver. The rear buffer B_r = 12100 lb_f acts against the sear. Release constraint between sear and receiver, perform momentum balance with coefficient of restitution e = 0, activate constraint between barrel B and sear SR, and restart integration with new velocities.

(6) t_5 (Figure 6.26): The barrel B and sear SR come to rest. The barrel drive force F_b is deactivated, the drive force F_f is activated, and the rear buffer force B_r is deactivated. Restart integration.

(7) t_6 (Figure 6.21): If automatic fire is to terminate, the barrel B and sear SR return to the initial sear position. The sear impacts the receiver, and the sear and barrel are captured by the receiver. Perform momentum balance

with coefficient of restitution e = 0 and activate the constraint between sear and receiver. The cycle is complete with sear and barrel locked to receiver.

(7') t_6' (Figure 6.21): If automatic fire is to continue, the barrel B and sear SR return to the initial sear position. The sear impacts the receiver and is captured by the receiver, while the barrel is released from the sear. Release the constraint between sear and barrel, perform momentum balance with coefficient of restitution e = 0, activate the constraint between sear SR and receiver, and restart integration with new velocities. The cycle is complete and the barrel is in the runout configuration for another round.

Logical times t_1 to t_6 depend upon the state of the system; the relative horizontal displacements and relative velocities between bodies of the system. Since the horizontal position, velocity, and acceleration of body centers-of-mass are state variables, logical times are expressed as functions of these variables.

The logical events are defined as follows:

(1) t_1: $x_2 - 34.26 = \ell^1 = 0$

(2) t_2: $x_2 - 36.75 = \ell^2 = 0$

(3) t_3: $x_2 - 34.26 = \ell^1 = 0$

(4) t_4: $x_2 - x_3 - 16 = \ell^3 = 0$

(5) t_5: $\dot{x}_2 = \ell^4 = 0$

(6) t_6: $x_2 - x_3 - 16 = \ell^3 = 0$

The six events t_1 to t_6 are thus defined by the four logical variables ℓ^1 to ℓ^4. In order to incorporate these event predictors into the numerical integration algorithm, the derivatives of the above equations, with appropriate initial conditions, are formulated and integrated along with the system equations of motion. Thus,

$$\dot{\ell}^1 = \dot{x}_2 \ , \qquad \ell^1(0) = -18.26$$
$$\dot{\ell}^2 = \dot{x}_2 \ , \qquad \ell^2(0) = -20.75$$
$$\dot{\ell}^3 = \dot{x}_2 - \dot{x}_3 \ , \qquad \ell^3(0) = 0$$
$$\dot{\ell}^4 = \ddot{x}_2 \ , \qquad \ell^4(0) = 40$$

The procedure for determining the complete system state precisely at logical times t_1 to t_6, identified by logical variables ℓ^1 to ℓ^4, is as follows. An appropriate time step is determined by the numerical integration algorithm based on the previous system state, polynomial predictor order, and error tolerance. Each logical variable in succession is predicted ahead in time, using this time step. If no logical variable is found to have passed through zero, the program advances the solution by the desired time step and the process is repeated. If one or more logical variables have passed through zero, the precise times at which the corresponding logical variables are zero are calculated by interpolation, using the polynomial predictor. A solution is then forced at the earliest logical time, indicating occurance of the first event. Control is then returned to user supplied subroutines so that actions can be taken according to the intent of the active logical variable.

Discontinuous events can be categorized according to the order of increasing difficulty as follows:

(1) Those requiring restart of the integration process only, such as when discontinuous forces act on or within the system. These forces are not considered to be impulsive in nature, thus only discontinuous accelerations result.

(2) Those requiring momentum balance due to impulsive external loads (impact between bodies and mass capture or release is excluded) with no supplemental restitution equations or constraint equation modification.

(3) Those requiring momentum balance due to impact between bodies and mass capture or release. Supplemental coefficient of restitution equations are appended to the momentum balance equations to achieve the desired velocity changes. Constraints are added or deleted, as needed to facilitate mass capture or release.

The six events t_1 to t_6 fall into the following three categories:

(1) t_1, t_3, t_5 - These events define discontinuous forces of relatively small magnitude, therefore only a restart of the integration procedure is required.

(2) t_2 - This event defines an externally applied impulsive load requiring a momentum balance and restart of the integration procedure.

(3) t_4, t_6 - These events define impulsive loading, due to impact between bodies of the system, and mass capture and release. Supplemental equations are required for momentum balance and a restart of the integration procedure is required.

The effects of the various events at logical times t_1 to t_6 on the position, velocity, and acceleration of the barrel are shown in Figures 6.27 to 6.29, respectively.

The logical events are clearly marked by discontinuities in acceleration of the gun tube as shown in Figure 6.29. At t_2, an impulse of - 880 lb_f-sec is applied to the barrel, resulting in a discontinuity in velocity and a change in barrel direction. The applied loads are changed at this instant, resulting in a net change in acceleration.

Impact between barrel and sear occurs at t_4, resulting in a second discontinuity in barrel velocity. Buffer action on the combined sear and barrel mass results in an acceleration of 4500 in/sec^2 in the time interval t_4 to t_5. The dynamic response curves of Figures 6.28 and 6.29 clearly indicate the intermittent nature of weapon mechanism motion. It is emphasized that one standard set of equations of motion and one standard set of momentum balance equations are generated by the program. At the various logical times, one simply appends or removes constraints, restitution equations, forces, impulses, etc., as directed by programmed control

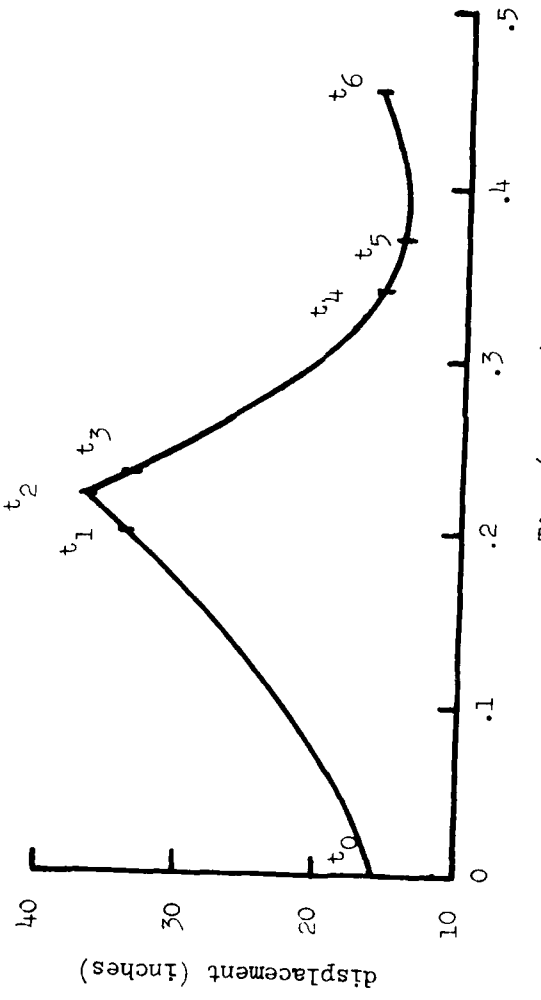


Figure 6.27. Horizontal displacement of the gun tube.

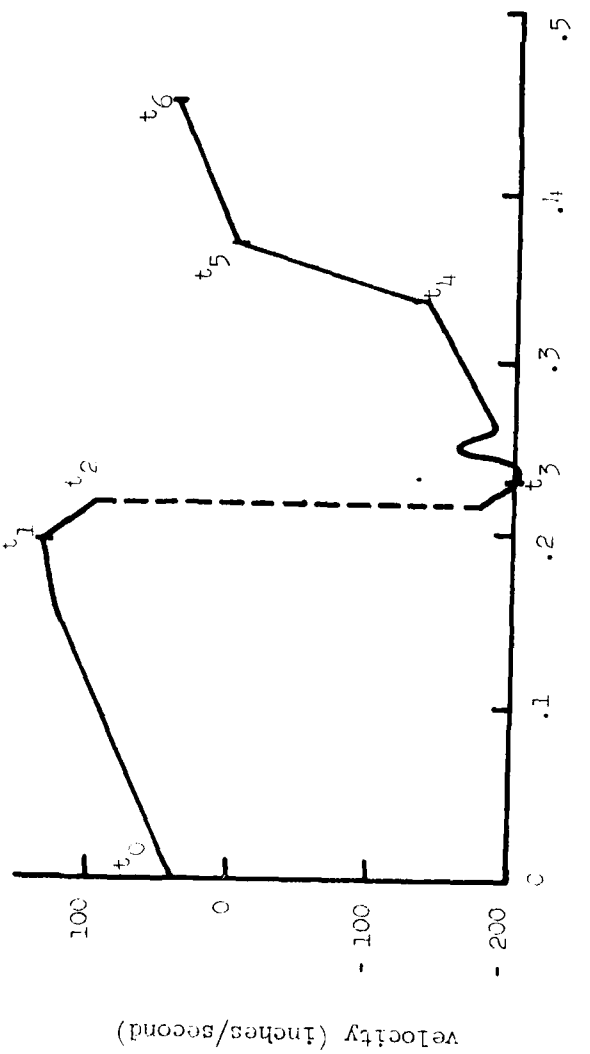


Figure 6.28. Horizontal velocity of the gun tube.

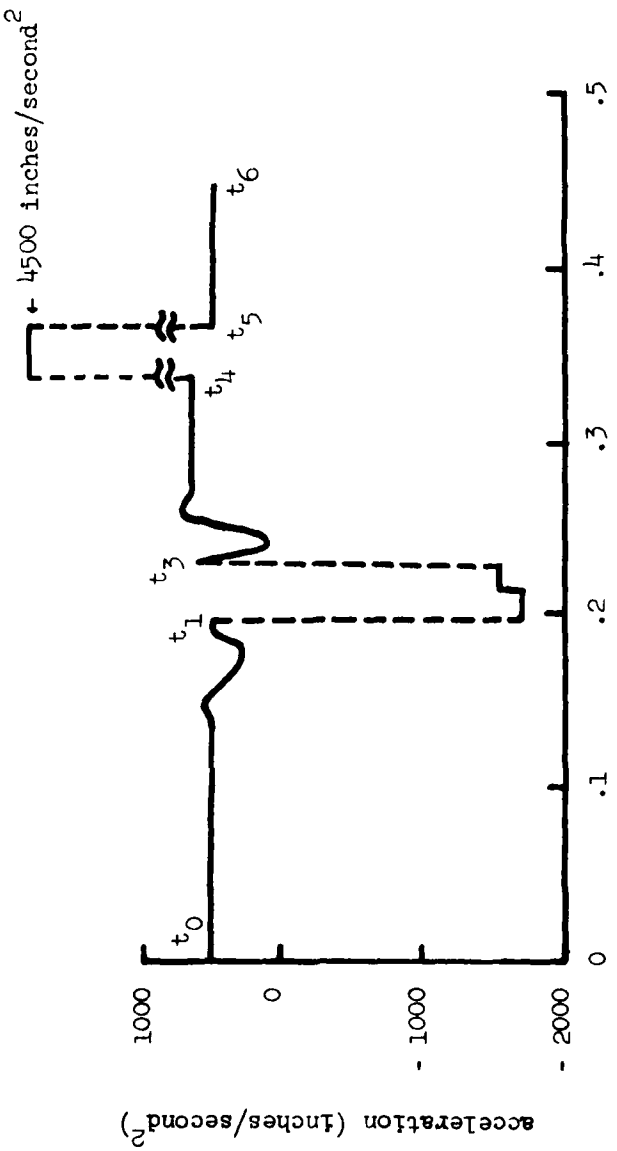


Figure 6.29. Horizontal acceleration of the gun tube.

logic, determined by the active logical variables. Thus, an entire simulation can be completed in a single computer run.

CHAPTER 7

CONCLUSIONS AND RECOMMENDATIONS

7.1 Conclusions

In this report, two methods are presented to improve efficiency of a general purpose dynamic analysis and design system computer program. The first method employs Gaussian elimination in the constraint Jacobian matrix to identify independent and dependent generalized coordinates, thus defining a minimal set of differential equations whose solution yields total system response. The second method employs pieced interval analysis with a logical events monitor and momentum balance capability to efficiently handle intermittent motion problems.

The analysis techniques have demonstrated an order of magnitude improvement in program performance. Since these methods require no change in equation formulation, generality of the program is maintained.

The methods developed are effectively applied to kinematic, kinetostatic, and dynamic analysis of constrained planar and three-dimensional systems. In addition, current investigations have demonstrated that performance of an analysis program that includes body flexibility is significantly improved when a minimum number of equations of motion are solved.

7.2 Recommendations

The method of coordinate partitioning has potential in other areas where coordinates are related by algebraic constraint equations. For example, Heltne [59] makes the following statements in his thesis when discussing techniques for solving large-scale nonlinear programming problems:

(1) "Luenberger [60] in his proof of the convergence rate of the GRG algorithm [59] shows this convergence rate to be coordinate dependent. That is, the convergence rate is a function of which variables are selected to be basic (dependent) and which nonbasic (independent)."

(2) "A second theoretical problem is the use of (matrix) structural considerations for selecting the basis. For example, the use of a relation such as maximizing the trace or the product of the diagonal elements of a matrix."

He further states that, "a nonzero diagonal does not imply a nonsingular matrix, but some relation based upon the elements of the Jacobian matrix may be a feasible alternative."

The method of Gaussian elimination with full row and column pivoting for coordinate partitioning accomplishes all of the above requirements. Furthermore, as discussed in Chapter 3, the resulting matrix will generally have the largest determinant of all possible submatrices, it will generally be well conditioned, and it has the largest elements on the diagonal.

The coordinate partitioning method has pitfalls when the constraint Jacobian matrix is sparse. To maintain program efficiency sparse matrix algorithms employ a partial pivoting strategy that generally does not select the largest pivotal elements. Thus, coordinate partitioning by this method will not be optimum. This may cause problems when attempting to maintain error control on dependent variables.

The approach taken in this study is to periodically construct a full matrix version of the constraint Jacobian and factor using the algorithm of Appendix A. This step, requiring $O(r^3)$ operations and $m \times 3n$ memory locations, is undesirable for large sparse systems. Following this, the sparse matrix is then augmented to force zero changes in independent variables at each iteration step. When systems experience large angular displacements the independent variable set changes frequently and correspondingly the sparse matrix structure changes. Costly symbolic sparse matrix refactorizations are then required [61].

Heltne [59] has recently developed sparse matrix manipulation algorithms employing block lower triangular factorization. This technique is also employed by the sparse matrix code in DADS. However, he has further developed an algorithm for ordering matrices after a rank-one update. That is, when the matrix has been put into block lower triangular form and the independent and dependent sets change, generally a new symbolic factorization can be obtained at a fraction of the cost of a complete matrix refactorization. He

has demonstrated an order of magnitude reduction in computer cost. When the variable sets change often, which is the case when bodies experience large angular displacements, the savings will be significant.

Additional research required in this and related areas is:

(1) Investigate the most efficient methods of Gaussian elimination employing sparse matrix techniques solely for the purpose of identifying optimum coordinate partitioning, thus eliminating full matrix factorization.

(2) Implement Heltne's matrix ordering rank-one update algorithm into the sparse matrix code.

(3) Investigate methods for determining when Gaussian elimination is required for the purpose of redetermining the partitioning of variables into dependent and independent sets.

(4) Investigate methods of analyzing coordinate velocities and accelerations as an alternate procedure for determining the need for applying Gaussian elimination in 3. Or as a complete replacement for 3. One such technique might be to compare differences between extrapolated dependent variables (provided by the integration algorithm) and the corrected dependent variables (provided by Newton iteration to satisfy constraints). A large discrepancy between a predicted and a corrected dependent variable is an indication that it should become

independent. Correspondingly an independent variable having a reduced or minimum velocity and an acceleration indicating that it may remain small for some time should then be made dependent. The reverse process of looking at maximum dependent velocity and corresponding acceleration could also be employed to identify a variable for the independent set.

(5) Investigate methods for relating constraint closure tolerance to numerical integration error tolerance. Tight constraint tolerance results in excessive Newton iterations with no gain in accuracy. Loose constraint tolerance results in poor dependent variable prediction which again results in additional corrector iterations or excessive dependent variable error. Methods that dynamically adjust constraint tolerance to minimize the number of corrector iterations should be investigated.

APPENDIX A

RANK DETERMINATION AND DECOMPOSITION OF SINGULAR MATRICES

A.1 Introduction

The technique and basic structure of the subprogram described here are patterned after the subroutine MFGR of the Scientific Subroutine Package, an IBM application program [22]. Consider a system of equations of the form

$$A_{(m \times n)} \cdot x_{(n \times 1)} = b_{(m \times 1)} \tag{A.1}$$

where in general the matrix A may lack full row and/or column rank. The task is to determine if one or more solutions to Equation A.1 exist, and to find at least one efficiently and accurately.

The following calculations can be performed on a general rectangular matrix which allows one to obtain solutions to Equation A.1 if such exist:

(1) Determine matrix rank and linearly independent rows and columns.

(2) Express a submatrix of maximal rank, as a product of triangular factors.

(3) Express nonbasic rows in terms of basic rows.

(4) Express basic variables in terms of free variables.

A.2 Theoretical Background

Basic variables, taken here as dependent variables, will be denoted by the vector u and free variables, taken as independent variables, will be denoted by the vector v.

Calculation (1) is most critical. Matrix rank is determined using the standard Gaussian elimination technique, with complete pivoting. Therefore, the rows and columns of the m by n matrix A are generally interchanged at each elimination step to bring the largest element to the pivot position. The interchange information is recorded in two integer permutation vectors IROW and ICOL.: The i^{th} row (column) of the interchanged matrix corresponds to the IROW $(I)^{th}$ row (ICOL $(I)^{th}$ column) in the original matrix A. Initially, IROW(J) = J and ICOL(J) = J.

The notation A^i is used for the interchanged matrix implied at the i^{th} elimination step. Superscripts do not mean powers. They indicate the current elimination step at which the result is obtained.

A.2.a First Elimination Step

Let a_{jk} be the absolutely greatest element of matrix A, which is found first in a columnwise scan. The internal tolerance TOL is set equal to $|EPS \cdot a_{jk}|$. The parameter EPS is a specified error level whose magnitude should be of the order of the existing computer round off error level.

If $|a_{jk}| \leq$ TOL, further calculation is bypassed. Otherwise rows 1 and j and columns 1 and k of matrix A are interchanged, giving A^1. The same interchanges must be applied to IROW, interpreted as a column vector, and to ICOL, interpreted as a row vector.

A^1 is uniquely expressed as the product $L^1 \cdot D^1 \cdot U^1$ by imposing the following conditions:

(I) L^1 is the m by m identity matrix, except for the first column. The first diagonal element has a value of one.

(II) D^1 is an m by n matrix with first diagonal element equal to one, while all remaining elements of the first row and column are equal to zero.

(III) U^1 is the n by n identity matrix, except for the first row.

$$A^1 = \begin{bmatrix} a_{11}^1 & A_{12}^1 \\ A_{21}^1 & A_{22}^1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ L_{21}^1 & I \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & D_{22}^1 \end{bmatrix} \begin{bmatrix} u_{11}^1 & U_{12}^1 \\ 0 & I \end{bmatrix}$$

$$= L^1 \cdot D^1 \cdot U^1 \quad ,$$

More explicitly,

$$
\begin{aligned}
a_{11}^{1} &= u_{11}^{1} \\
A_{12}^{1} &= U_{12}^{1} \\
A_{21}^{1} &= L_{21}^{1} \cdot u_{11}^{1} \\
A_{22}^{1} &= L_{21}^{1} \cdot U_{12}^{1} + D_{22}^{1}
\end{aligned}
\qquad \text{(A.2)}
$$

Capital letters are used as notation for matrices in this appendix. Small letters represent scaler or vector quantities. The symbol 0 means zero matrices and I means identity matrices, where dimensions are implied by compatibility. The terms in Equation A.2 are further described as follows:

The elements of the first column of L^1 are

$$\ell_{11}^{1} = 1,\ \ell_{21}^{1} = a_{21}^{1}/a_{11},\ldots,\ \ell_{m1}^{1} = a_{m1}^{1}/a_{11}^{1}$$

The elements of submatrix D_{22}^{1} of D^1 are

$$d_{ik}^{1} = a_{ik}^{1} - \ell_{il}^{1} \cdot u_{lk}^{1} = a_{ik}^{1} - \frac{a_{il}^{1} \cdot a_{lk}^{1}}{a_{11}^{1}} \quad ; \quad \begin{matrix} i = 2, \ldots, m \\ k = 2, \ldots, n \end{matrix}$$

The elements of the first row of U^1 are

$$u_{11}^{1} = a_{11}^{1},\ u_{12}^{1} = a_{12}^{1},\ \ldots,\ u_{1n}^{1} = a_{1n}^{1}$$

Note that it is possible to record all nontrivial entries of L^1, D^1, and U^1 compactly in the storage locations occupied by the original A, storing only:

$$\begin{bmatrix} u_{11}^1 & U_{12}^1 \\ L_{21}^1 & D_{22}^1 \end{bmatrix}$$

A.2.b. Second Elimination Step

Let d_{jk}^1, $\begin{matrix} j \geqq 2 \\ k \geqq 2 \end{matrix}$ be the absolutely largest element of D_{22}^1. If $|d_{jk}^1| \leqq$ TOL, D_{22}^1 is interpreted as being the zero matrix.

If D_{22}^1 is not zero in the above sense, it may be decomposed analogously. In the compact scheme of above, rows 2 and j, and columns 2 and k are interchanged, obtaining

$$\begin{bmatrix} u_{11}^1 & U_{12}^2 \\ L_{21}^2 & D_{22}^2 \end{bmatrix}$$

The same interchanges must be performed with IROW (interpreted as a column vector), with ICOL (interpreted as a row vector), and with A^1 giving

$$A^2 = \begin{bmatrix} 1 & 0 \\ L_{21}^2 & I \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & D_{22}^2 \end{bmatrix} \begin{bmatrix} u_{11}^1 & U_{12}^2 \\ 0 & I \end{bmatrix}$$

Now D_{22}^2 may be expressed uniquely as the product of the form L·D·U, imposing conditions I, II, and III. The result is

$$D_{22}^2 = \begin{bmatrix} 1 & 0 \\ L_{32}^2 & I \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & D_{33}^2 \end{bmatrix} \begin{bmatrix} u_{22}^2 & U_{23}^2 \\ 0 & I \end{bmatrix}$$

Then $A^2 = L^2 \cdot D^2 \cdot U^2$, with

$$L^2 = \begin{bmatrix} 1 & 0 & 0 \\ \ell_{21}^2 & 1 & 0 \\ L_{31}^2 & L_{32}^2 & I \end{bmatrix}$$

$$D^2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & D_{33}^2 \end{bmatrix}$$

$$U^2 = \begin{bmatrix} u_{11}^1 & u_{12}^2 & U_{13}^2 \\ 0 & u_{22}^2 & U_{23}^2 \\ 0 & 0 & I \end{bmatrix}$$

where

$$L_{21}^2 = \begin{bmatrix} \ell_{21}^2 \\ L_{31}^2 \end{bmatrix}, \quad U_{12}^2 = [u_{12}^2, U_{13}^2]$$

A.2.c. Final Result of Elimination Process

At the next elimination step D_{33}^2 is factorized, and so on. Now assume that finally $D_{r+1,r+1}^r$ equals zero in the sense that all its elements are absolutely less than TOL. This means that A has the rank r and the end result is the factorization $A^r = L^r \cdot D^r \cdot U^r$; that is,

$$A^r = \begin{bmatrix} 1 & 0 & \ldots 0 & 0 \\ \ell_{21}^2 & 1 & . \ 0 & 0 \\ . & . & . \ . & . \\ . & . & . & . \\ . & . & .. & . \\ \ell_{r1}^r & \ell_{r2}^r & \ldots 1 & 0 \\ L_{r+1,1}^r & L_{r+1,2} \ldots & L_{r+1,r}^r & I \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0\ldots0 & 0 \\ 0 & 1\ldots0 & 0 \\ \vdots & \vdots\ddots\vdots & \vdots \\ 0 & 0\ldots1 & 0 \\ 0 & 0\ldots0 & D^{r}_{r+1,r+1} \end{bmatrix}$$

$$\begin{bmatrix} u^{1}_{1} & u^{2}_{12} \ldots & u^{r}_{1r} & U^{r}_{1,r+1} \\ 0 & u^{2}_{22} \ldots & u^{r}_{2r} & U^{r}_{2,r+1} \\ \vdots & \ddots & \vdots & \vdots \\ 0 & 0 \ldots & u^{r}_{rr} & U^{r}_{r,r+1} \\ 0 & 0 \ldots & 0 & I \end{bmatrix} \qquad (A.3)$$

Neglecting the small elements in $D^{r}_{r+1,r+1}$ this may be written more compactly as:

$$A^{r} = \begin{bmatrix} L \\ LR \end{bmatrix} [U, UR]$$

with $$L = \begin{bmatrix} 1 & 0 \ldots 0 \\ \ell^{2}_{21} & 1 \quad\quad 0 \\ \vdots & \ddots \quad \vdots \\ \ell^{r}_{r1} & \ell^{r}_{r2} \ldots 1 \end{bmatrix}$$

$$LR = [L^{r}_{r+1,1}, L^{r}_{r+1,2}, \ldots, L^{r}_{r+1,r}]$$

$$U = \begin{bmatrix} u_{11}^{1} & u_{12}^{2} & \cdots & u_{1r}^{r} \\ 0 & u_{22}^{2} & \cdots & \vdots \\ \vdots & & \ddots & u_{rr}^{r} \\ 0 & 0 & & \end{bmatrix}$$

$$UR = \begin{bmatrix} U_{1,r+1}^{r} \\ U_{2,r+1}^{r} \\ \vdots \\ U_{r,r+1}^{r} \end{bmatrix}$$

The matrix L is of dimension r by r and unit lower triangular and U is of dimension r by r and upper triangular. The matrix LR is of dimension m - r by r; if the given matrix A is row regular (that is, m = r) LR is absent in the final factorization. The matrix UR is of dimension r by n - r; if the given matrix A is column regular (that is, n = r) UR is absent in the final factorization.

A.2.d. Further Calculations Performed

The problem of matrix factorization arises in connection with the solution of systems of equations Ax = b. Three different cases must be distinguished, as follows:

(1) r = m = n;

A is nonsingular, so Ax = b has a uniquely determined solution.

(2) $r < m$;

A is not row regular, so solutions of $Ax = b$ exist only if linear combinations among the rows of A are also valid among the rows of b

(3) $r < n$;

A is not column regular, so $Ax = 0$ has nontrivial solutions.

The cases (2) and (3) may occur combined.

The solution, if it exists, is uniquely determined if $r = n$. Otherwise, it contains $n - r$ free parameters. Often one requires the linear combinations among the rows of a given matrix A and the linear forms expressing basic variables in terms of free ones. Therefore, instead of LR and UR, matrices C and H are returned, containing linear combinations and homogeneous solutions respectively.

Observe that the above calculated factorization belongs to the interchanged matrix A^r. Therefore, $A^r x^r = b^r$ is dealt with instead of $Ax = b$, where $\begin{bmatrix} x^r \\ b^r \end{bmatrix}$ is obtained from $\begin{bmatrix} x \\ b \end{bmatrix}$, using the $\begin{bmatrix} \mathrm{ICOL}(I)^{th} \\ \mathrm{IROW}(J)^{th} \end{bmatrix}$ element of $\begin{bmatrix} x \\ b \end{bmatrix}$ as the $\begin{bmatrix} I^{th} \\ J^{th} \end{bmatrix}$ element of

$$\begin{bmatrix} x^r & I = 1, \ldots, n \\ b^r & ; \quad J = 1, \ldots, m \end{bmatrix} .$$

Let x^r, b^r be partitioned into $\begin{bmatrix} u \\ v \end{bmatrix}$ and $\begin{bmatrix} b^1 \\ b^2 \end{bmatrix}$. Then,

$$\begin{bmatrix} L \\ LR \end{bmatrix} \cdot [U, UR] \cdot \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} b^1 \\ b^2 \end{bmatrix}$$

or explicity,

$$L \cdot U \cdot u + L \cdot UR \cdot v = b^1$$

$$LR \cdot U \cdot u + LR \cdot UR \cdot v = b^2$$

Since L and U are nonsingular, this implies that

$$u = U^{-1} \cdot L^{-1} \cdot b^1 - U^{-1} \cdot UR \cdot v$$

$$LR \cdot L^{-1} \cdot b^1 = b^2$$

For convenience, LR is replaced by $LR \cdot L^{-1} = C$ and UR is replaced by $- U^{-1} \cdot UR = H$, while L and U remain untouched. Consistency requires that $b^2 = C \cdot b^1$, and homogeneous solutions are given by $u = H \cdot v$. In case of a consistent system of equations $A^r \cdot x^r = b^r$, the general solution is $x^r = \begin{bmatrix} u \\ v \end{bmatrix}$, with $u = U^{-1} \cdot L^{-1} \cdot b^1 + H \cdot v$, while the values of the free variables contained in v may be chosen arbitrarily.

A.2.e. Format of Results

The subprogram returns matrices L, U, C, H, and D in the storage area originally occupied by the input matrix A, in the compact scheme shown in Figure A.1.

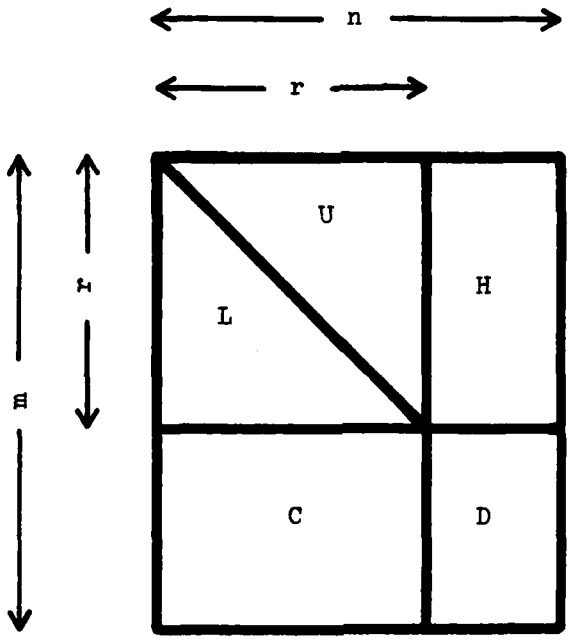


Figure A.1. Format scheme of the processed matrix.

REFERENCES

1. Hrones, J. A. and Nelson, G. L., Analysis of the Four-Bar Linkage, Technology Press of M.I.T., Cambridge, MA, and Wiley, New York, NY, 1951.

2. Artobolevskii, I. I., Mechanisms for the Generation of Plane Curves, (trans. by R. D. Wills), Macmillan, New York, NY, 1964.

3. General Motors, "GMR DYANA" Programming Manuals I and II, GMR 229 and GMR 237, Research Laboratories, G. M. Corp., Detroit, MI, 1959.

4. Korn, G. A. and Korn, T. M., Electronic Analog and Hybrid Computers, McGraw-Hill, Inc., New York, NY, 1964.

5. Okuno, S., Simulation of Motor Scraper Overturns, Ph.D. dissertation, Cornell University, August, 1977.

6. Davis, D., Simulation and Model Verification of Agricultural Tractor Overturns, Ph.D. dissertation, Cornell University, 1973.

7. Paul, B. and Krajcinovic, D., "Computer Analysis of Machines With Planar Motion--Part I: Kinematic; Part II: Dynamics", ASME Journal of Applied Mechanics, Ser. E, Vol. 37, 697-712, 1970.

8. Chace, M. A. and Smith, D. A., DAMN-A Digital Computer Program for the Dynamic Analysis of Generalized Mechanical Systems, SAE paper 710244, January, 1971.

9. Sheth, P. N. and Uicker, J. J., Jr., "IMP (Integrated Mechanisms Program), A Computer Aided Design Analysis System for Mechanisms and Linkages", ASME Journal of Engineering for Industry, Ser. B, Vol. 94, 454-464, 1972.

10. Orlandea, N., Chace, M. A., and Calahan, D. A., "A Sparsity-Oriented Approach to the Dynamic Analysis and Design of Mechanical Systems, Parts I and II", ASME Journal of Engineering for Industry, Ser. B, Vol. 99, 773-784, 1977.

11. Langrana, N. A. and Bartel, D. L., "An Automated Method for Dynamic Analysis of Spatial Linkages for Biomedical Applications",

ASME Journal of Engineering for Industry, Ser. B, Vol. 97, No. 2, 566-574, 1975.

12. Rogers, R. J. and Andrews, G. C., "The Simulation of Planar Systems Using a Simplified Vector-Network Method", Mechanism and Machine Theory, Vol. 10, 509-519, 1975.

13. Liegois, A., Khalil, W., and Dumas, J. M., "Mathematical and Computer Models of Interconnected Mechanical Systems", Proceedings of the Symposium on Theory and Practice of Robots and Manipulators, Warsaw, Poland, September, 1976.

14. Williams, R. J. and Seireg, A., "Interactive Modeling and Analysis of Open or Closed Loop Dynamic Systems With Redundant Actuators", ASME Journal of Mechanical Design, Vol. 101, 407-416, July, 1979.

15. Paul, B., Kinematics and Dynamics of Planar Machinery, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1979.

16. Sheth, P. N., A Digital Computer Based Simulation Procedure for Multiple Degree of Freedom Mechanical Systems with Geometric Constraints, Ph.D. dissertation, University of Wisconsin, 1972.

17. Smith, D. A., Reaction Forces and Impact in Generalized Two-Dimensional Mechanical Dynamic Systems, Ph.D. dissertation, The University of Michigan, 1971.

18. Shampine, L. F., Watts, H. A. and Davenport, S. M., "Solving Nonstiff Ordinary Differential Equations - the State of the Art", SIAM Review, Vol. 18, No. 3, 376-411, July, 1976.

19. Orlandea N., Node Analogous, Sparsity-Oriented Methods for Simulation of Mechanical Dynamic Systems, Ph.D. dissertation, The University of Michigan, 1973.

20. Gear, C. W., "Simultaneous Numerical Solution of Differential-Algebraic Equations", IEEE Transactions on Circuit Theory, Vol. CT-18, No. 1, 89-95, January, 1971.

21. Haug, E. J. Wehage, R. A., and Barman, N. C., "Design Sensitivity Analysis of Planar Mechanism and Machine Dynamics", to appear in ASME Journal of Mechanical Design.

22. System/360 Scientific Subroutine Package (360A-CM-03X), Programmer's Manual (H20-0205-4), 1970.

23. Orlandea, N., Wiley, J. C. and Wehage, R. A., ADAMS2: A Sparse Matrix Approach to the Dynamic Simulation of Two-Dimensional Mechanical Systems, Society of Automotive Engineers Technical Paper Series 780486, 1978.

24. Nikravesh, P. E. and Chung, I. S., "Application of Euler Parameters to the Dynamic Analysis of Three Dimensional Constrained Mechanical Systems - A General Purpose Computer Program", submitted to ASME Journal of Mechanical Design.

25. Song, J. O. and Haug, E. J., "Dynamic Analysis of Flexible Mechanisms", to appear in Computer Methods in Applied Mechanics and Engineering.

26. Chao, E. Y. and Rim, K., "Application of Optimization Principles in Determining the Applied Moments in Human Leg Joints During Gait", Journal of Biomechanics, Vol. 6, 497-510, 1973.

27. Szidarovszky, F. and Yakowitz, S., Principles and Procedures of Numerical Analysis, Plenum Press, 227 West 17th Street, New York, NY.

28. Shampine, L. F. and Gordon, M. K., Computer Solution of Ordinary Differential Equations: The Initial Value Problem, W. J. Freeman, San Francisco, CA, 1975.

29. Carver, M. B., "Efficient Integration Over Discontinuities in Ordinary Differential Equation Simulations", Mathematics and Computers in Simulation, Vol. 20, 190-196, 1978.

30. Dugas, R., A History of Mechanics, Editions du Gri-fon, Neuchatel-Switzerland, 1955.

31. van Bokhoven, W. M. G., "Linear Implicit Differentiation Formulas of Variable Step and Order", IEEE Transactions on Circuits and Systems, CAS-22, No. 2, 109-115, February, 1975.

32. Duff, I. S., MA28-A Set of FORTRAN Subroutines for Sparse Unsymmetric Linear Equations, Report No. AERE-R.8730, Computer Science and Systems Division, AERE Harwell, Oxfordshire, July, 1977.

33. Greenwood, D. T., Principles of Dynamics, Prentice-Hall, Englewood Cliffs, NJ, 1965.

34. Haug, E. J. and Arora, J. S., Applied Optimal Design, John Wiley and Sons, New York, NY, 1979.

35. Chua, L. O. and Lin, P. M., Computer-Aided Analysis of Electronic Circuits: Algorithm and Computational Techniques, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1975.

36. Gear, C. W., Numerical Initial Value Problems in Ordinary Differential Equations, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1971.

37. Brameller, A., Allen, R. N., and Hamen, Y. M., Sparsity, Pitman Publishing Corporation, New York, NY, 1976.

38. Friend, J. H., Guralnik, D. B., et al., Webster's New World Dictionary of the American Language, College Edition, The World Publishing Company, Cleveland, OH, 1959.

39. Wells, D. A., Shaum's Outline of Theory and Problems of Lagrangian Dynamics, McGraw-Hill Book Co., St. Louis, MO, 1967.

40. Sheth, P. N., Improved Iterative Techniques for the (4 x 4) Matrix Method of Kinematic Analysis, M. S. dissertation, University of Wisconsin, 1968.

41. Forsythe, G. E. and Moler, C. B., Computer Solution of Linear Algebraic Systems, Prentice-Hall, Englewood Cliffs, NJ, 1967.

42. Paul, B., "Analytical Dynamics of Mechanisms--A Computer Oriented Overview", Mechanism and Machine Theory, Vol. 10, 481-507, 1975.

43. Huang, R. C., Haug, E. J. and Andrews, J. G., "Sensitivity Analysis and Optimal Design of a Mechanical System with Intermittent Motion", ASME Journal of Mechanical Design, Vol. 100, 492-499, 1978.

44. Nordsieck, A., "Automatic Numerical Integration of Ordinary Differential Equations", Proceedings of Symposium on Applied Math, Vol. 15, 241-250, 1963.

45. Brayton, R. K., Gustavson, F. G., and Hachtel, G. D., "A New Efficient Algorithm for Solving Differential-Algebraic Systems Using Implicit-Backward Differential Formulas", Proceedings of the IEEE, Vol. 60, No. 1, 98-108, January, 1972.

46. Calahan, D. A. and Orlandea, N., "A Program for the Analysis and Design of General Dynamic Mechanical Systems", AFIPS Conference Proceedings, Vol. 41, Part II, 885-888, 1972.

47. Beck, R. R. and Kamm, I. O., Cybernetically Coupled Research Vehicle, Society of Automotive Engineers, Rep. 75-217, February, 1975.

48. Kamm, I. O. and Beck, R. R., The Performance of the Coupled M-113 Armored Personnel Carriers, 5th International Conference, Detroit, Michigan, International Society for Terrain-Vehicle Systems, June, 1975.

49. Conte, S. D. and de Boor, C., Elementary Numerical Analysis an Algorithmic Approach, McGraw-Hill Book Co., St. Louis, MO, 1972.

50. Vandergraft, J. S., Introduction to Numerical Computations, Academic Press, New York, NY, 1978.

51. Vehicle platform stability analysis being performed for the U. S. Army TARADCOM.

52. Hoogterp, F. B., Digital/Analog Vehicle Ride Simulation, Technical Report No. 11705, U. S. Army Tank Automotive Command, Warren, MI, November, 1972.

53. Beck, R. R. and Wehage, R. A., "The Modeling and Simulation of Two Coupled M-113 Armored Personnel Carriers", Proceedings of the Tenth Annual Pittsburgh Conference on Modeling and Simulation, Vol. 10, Part 2, 353-359, 1979.

54. Ehle, P. E. and Haug, E. J., "A Logical Function Method for Dynamic and Design Sensitivity Analysis of Mechanical Systems with Intermittent Motion", to appear in ASME Journal of Mechanical Design.

55. Kecs, W. and Teodorescu, P. P., Applications of the Theory of Distributions in Mechanics, Abacus Press, Tunbridge Wells, Kent, England, 1974.

56. Huang, R. C. and Haug, E. J., "A Distribution Theoretic Method for Design Sensitivity Analysis of Mechanisms with Intermittent Motion; Part I: Theory; Part II: Applications", Developments in Mechanics, Proceedings of the 16th Midwestern Mechanics Conference, Vol. 10, 115-126, 1979.

57. Ehle, P. E., Dynamic Analysis and Design Sensitivity Analysis of Mechanisms with Intermittent Motion, Technical Report No. 48, Division of Materials Engineering, College of Engineering, The University of Iowa, February, 1979.

58. Wehage, R. A. and Haug, E. J., "Generalized Coordinate Partitioning for Dimension Reduction in Analysis of Constrained Dynamic Systems", to appear in the ASME Journal of Mechanical Design.

59. Heltne, D. R., Nonlinear Programming in Chemical Engineering Design, Ph.D. dissertation, The University of Iowa, 1979.

60. Luenberger, D. G., Introduction to Linear and Nonlinear Programming, Addison-Wesley, Reading, MA, 1973.

61. Reid, J. K., Sparse Matrices, C. S. S. 31, Presented at the IMA Conference, "The State-of-the-Art in Numerical Analysis", York, April 1976, Computer Science and Systems Division, A. E. R. E., Harwell, Didcot, Oxen, May, 1976.

DISTRIBUTION LIST

Please notify USATACOM, DRSTA-ZSA, Warren, Michigan 48090, of corrections and/or changes in address.

Commander (25)
US Army Tk-Autmv Command
R&D Center
Warren, MI 48090

Superintendent (02)
US Military Academy
ATTN: Dept of Engineering
Course Director for
Automotive Engineering

Commander (01)
US Army Logistic Center
Fort Lee, VA 23801

US Army Research Office (02)
P.O. Box 12211
ATTN: Dr. F. Schmiedeshoff
Dr. R. Singleton
Research Triangle Park, NC 27709

HQ, DA (01)
ATTN: DAMA-AR
Dr. Herschner
Washington, D.C. 20310

HQ, DA (01)
Office of Dep Chief of Staff
for Rsch, Dev & Acquisition
ATTN: DAMA-AR
Dr. Charles Church
Washington, D.C. 20310

HQ, DARCOM
5001 Eisenhower Ave.
ATTN: DRCDE
Dr. R.L. Haley
Alexandria, VA 22333

Director (01)
Defense Advanced Research
Projects Agency
1400 Wilson Boulevard
Arlington, VA 22209

Commander (01)
US Army Combined Arms Combat
Developments Activity
ATTN: ATCA-CCC-S
Fort Leavenworth, KA 66027

Commander (01)
US Army Mobility Equipment
Research and Development Command
ATTN: DRDME-RT
Fort Belvoir, VA 22060

Director (02)
US Army Corps of Engineers
Waterways Experiment Station
P.O. Box 631
Vicksburg, MS 39180

Commander (01)
US Army Materials and Mechanics
Research Center
ATTN: Mr. Adachi
Watertown, MA 02172

Director (03)
US Army Corps of Engineers
Waterways Experiment Station
P.O. Box 631
ATTN: Mr. Nuttall
Vicksburg, MS 39180

Director (04)
US Army Cold Regions Research
& Engineering Lab
P.O. Box 282
ATTN: Dr. Freitag, Dr. W. Harrison
Dr. Liston, Library
Hanover, NH 03755

President (02)
Army Armor and Engineer Board
Fort Knox, KY 40121

Commander (01)
US Army Arctic Test Center
APO 409
Seattle, WA 98733

Commander (02)
US Army Test & Evaluation
Command
ATTN: AMSTE-BB and AMSTE-TA
Aberdeen Proving Ground, MD
21005

Commander (01)
US Army Armament Research
and Development Command
ATTN: Mr. Rubin
Dover, NJ 07801

Commander (01)
US Army Yuma Proving Ground
ATTN: STEYP-RPT
Yuma, AZ 85364

Commander (01)
US Army Natic Laboratories
ATTN: Technical Library
Natick, MA 01760

Director (01)
US Army Human Engineering Lab
ATTN: Mr. Eckels
Aberdeen Proving Ground, MD
21005

Director (02)
US Army Ballistic Research Lab
Aberdeen Proving Ground, MD
21005

Director (02)
US Army Materiel Systems
Analysis Agency
ATTN: AMXSY-CM
Aberdeen Proving Ground, MD
21005

Director (02)
Defense Documentation Center
Cameron Station
Alexandria, VA 22314

US Marine Corps (01)
Mobility & Logistics Division
Development and Ed Command
ATTN: Mr. Hickson
Quantico, VA 22134

Keweenaw Field Station (01)
Keweenaw Research Center
Rural Route 1
P.C. Box 94-D
ATTN: Dr. Sung M. Lee
Calumet, MI 49913

Naval Ship Research & (02)
Dev Center
Aviation & Surface Effects Dept
Code 161
Washington, D.C. 20034

Director (01)
National Tillage Machinery Lab
Box 792
Auburn, AL 36830

Director (02)
USDA Forest Service Equipment
Development Center
444 East Bonita Avenue
San Dimes, CA 91773

Engineering Societies (01)
Library
345 East 47th Street
New York, NY 10017

Dr. I.R. Erlich (01)
Dean for Research
Stevens Institute of Technology
Castle Point Station
Hoboken, NJ 07030

Grumman Aerospace Corp (02)
South Oyster Bay Road
ATTN: Dr. L. Karafiath
Mr. F. Markow
M/S A08/35
Bethpage, NY 11714

Dr. Bruce Liljedahl (01)
Agricultural Engineering Dept
Purdue University
Lafayette, IN 46207

Mr. H.C. Hodges (01)
Nevada Automotive Test Center
Box 234
Carson City, NV 89701

Mr. R.S. Wismer (01)
Deere & Company
Engineering Research
3300 River Drive
Moline, IL 61265

Oregon State University (01)
Library
Corvallis, OR 97331

Southwest Research Inst (01)
8500 Culebra Road
San Antonia, TX 78228

FMC Corporation (01)
Technical Library
P.O. Box 1201
San Jose, CA 95108

Mr. J. Appelblatt (01)
Director of Engineering
Cadillac Gauge Company
P.O. Box 1027
Warren, MI 48090

Chrysler Corporation (02)
Mobility Research Laboratory,
Defense Engineering
Department 6100
P.O. Box 751
Detroit, MI 48231

CALSPAN Corporation (01)
Box 235
Library
4455 Benesse Street
Buffalo, NY 14221

SEM, (01)
Forsvaretsforskningsanstalt
Avd 2
Stockholm 80, Sweden

Mr. Hedwig (02)
RU III/6
Ministry of Defense
5300 Bonn, Germany

Foreign Science & Tech (01)
Center
220 7th Street North East
ATTN: AMXST-GEI
Mr. Tim Nix
Charlottesville, VA 22901

General Research Corp (01)
7655 Old Springhouse Road
Westgate Research Park
ATTN: Mr. A. Viilu
McLean, VA 22101

Commander (01)
US Army Developmant and
Readiness Command
5001 Eisenhower Avenue
ATTN: Dr. R.S. Wiseman
Alexandria, VA 22333